Technische Universität Ilmenau
Fakultät für Mathematik
und Naturwissenschaften
Institut für Mathematik
http://www.mathematik.tu-ilmenau.de/Math-Net/index_de.html

Postfach 10 05 65
D - 98684 Ilmenau
Germany
Tel.: 03677/69 3267
Fax: 03677/69 3272
Telex: 33 84 23 tuil d.
email: werner.neundorf@tu-ilmenau.de

Preprint No. M 08/03

Lösungsmethoden mit Maple

Betrachtung von Fehlern und Kondition sowie Darstellungsmöglichkeiten

Werner Neundorf

April 2003

[‡]MSC (2000): 65-01, 65-04, 65F05, 65F10, 65Y99, 68U99, 68Q25

Zusammenfassung

Successful application of numerical solvers depends not only on fully-developed software tools, but during the primary phase of solution there is necessary a wide basic education which includes the ability for problem modelling and analysis, possibilities of transforming the problem into another form and the choice of appropriate solution methods. This knowledge level can lead to higher quality of methods and computer programs, which are based on the algorithms. On the other hand, software tools, too, dispose of tutorial components and they can promote deeper reflections and new powerful ideas in varied ways.
Observations and experiences have shown, that it is not necessary to confront students of the first semesters studying natural sciences and technology with big shells of very extensive software tools. Instead, limited programs prove as effective and modern didactic means. They allow especially the far-reaching changes of configuration and in this way some possibilities of doing cognitive manipulations with regard to graphical and algebraic-symbolic elements, presentation and scope of results, selection of functions and methods, interactive mode of operation and repetition.
Computers and software belong to the cognitive technologies and can play an important role in the creation and development of mental models.
This is supported by selected examples and demonstrations from different areas like solving linear systems, matrix condition, dilemma of subtraction, transformation of expression, error propagation, recursive algorithms, approximate numbers, special computer effects and geometric observations.
The most calculations are done in Maple 8, some in the programming languages Turbo Pascal and Pascal-XSC. By itself or when coupled with other computing environments and Computer Algebra Systems, Maple can be incorporated effectively into the curriculum to enhance the understanding of both fundamental and advanced topics, while enabling the student actively to put theory into practice.
The aim is to show how you can write simple programs in Maple for doing numerical calculations, linear algebra, and programs for simplifying or transforming symbolic expressions, polynomials or mathematical formulas. It is assumed that the reader is familiar with using Maple interactively.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Es gibt viele Möglichkeiten, mathematische und andere Aufgabenstellungen in Ersatzprobleme umzuformen, die dann praktisch und mit Computereinsatz gelöst werden können. Dabei zeigt sich, dass die erfolgreiche Anwendung von Verfahren nicht nur von ausgereiften Softwaretools abhängt, sondern in der ersten Phase der Lösung eine breite fachliche Grundbildung für Problemanalyse, für Möglichkeiten der Umformung einer Aufgabe in eine andere adäquate Darstellung sowie für die Auswahl von entsprechenden Lösungsmethoden notwendig ist. Dieser Kenntnisstand kann zu einer höheren Qualität der Verfahren und der darauf aufbauenden Computerprogrammen führen. Andererseits verfügen auch die Softwarewerkzeuge über zahlreiche tutorielle Elemente und sie können in vielfältiger Weise zu tiefergreifenden Überlegungen und neuen Ideen führen. Hier bieten sich insbesondere der Einsatz von Computeralgebrasystemen (CAS) wie Maple (vgl. [9], [10]), Matlab (vgl. [20] – [23]), *Mathematica* oder Derive an.
Beobachtungen und Erfahrungen zeigen, dass es für Studenten der ersten Semester in naturwissenschaftlichen und technischen Fachrichtungen nicht unbedingt notwendig ist, das große Szenarium eines noch größeren Tools nahezubringen. Vielmehr erweisen sich überschaubare Programme als wirksame und moderne didaktische Werkzeuge. Sie gestatten insbesondere die weitgehende Konfigurierbarkeit und damit kognitive Manipulationsmöglichkeiten, auch bezüglich grafischer und algebraisch-symbolischer Elemente, Darstellungsweisen und Umfang der Ergebnisse, Funktions- und Methodenauswahl, interaktive Arbeitsweise und Wiederholbarkeit (vgl. [37]).

Auswahl einiger Softwareaspekte als methodisch-didaktische Hilfsmittel in kognitiven Modellen:
- Multiple-window Darstellungen,
- Desk-Top-Publishing Systeme,
- Veranschaulichung des Stellenwertsystems und Operationen darin,
- Computergrafik,
- Nutzung von Gleichungen und Funktionsgrafen,

- algebraische, geometrische und grafische Symbolsysteme,
- Darstellung von Formeln und Tabellen,
- Datenspeicherung und -manipulation,
- Wiederholbarkeit, Reflexion, Interaktion,
- Verstärkeraspekt, Reorganisationsaspekt.

Während rein rechnerisch-technische Aufgabenteile mit Hilfe des Computers sehr oft gelöst werden können, gewinnen die anspruchsvolleren Aspekte an Bedeutung. Dazu gehören natürlich die geschickte Umsetzung einer Anwendungssituation in ein passendes (mathematisches) Modell sowie die kreative Entwicklung und sachgemäße Beurteilung und Begründung mathematischer Begriffe und Verfahren.
Zu einigen ausgewählten Aufgabentypen werden im Kapitel 3 verschiedene Methoden vorgestellt sowie entsprechende Möglichkeiten, schöne Effekte, aber auch Besonderheiten und Grenzen bei der Nutzung von Programmen und CAS diskutiert.

Handelt es sich bei den zu lösenden Problemen um akademische oder solcher mit moderater Größenordnung, so bietet sich der Einsatz des **CAS Maple 8** an. Die symbolischen Rechnungen, die unter Verwendung einer exakten Arithmetik auch exakt durchgeführt werden, sind dann mit vertretbarem Zeitaufwand machbar. Jedoch kann z. B. eine wiederholte symbolische Matrix-Vektor-Multiplikation, wie sie in der linearen Algebra ständig auftritt, für große Felddimensionen erhebliche Zeit kosten. Hier und bei typischen numerischen Algorithmen ist dann die Verwendung der Gleitpunktarithmetik zu empfehlen, wie das in Matlab geschieht.
Die CAS erfahren eine ständige Weiterentwicklung und werden immer größer und leistungsfähiger. Programme und Arbeitsblätter, die unter älteren Versionen entstanden und gelaufen sind, sollte man bezüglich der Möglichkeiten und Veränderungen in neuen Versionen stets kritisch begutachten und eventuell anpassen. Das betrifft auch die breite Palette der Menüfunktionen. Unter diesem Aspekt sind auch die Informationen und Arbeitsblätter unter Maple V in [9] und [10] zu sehen und gegebenenfalls für Maple 8 zu modifizieren.
Das Preprint M09/95 *Kondition eines Problems und angepaßte Lösungsmethoden* [37] bietet genügend Material für die nun folgenden weitergehenden Untersuchungen mit Maple 8. Einige Betrachtungen sind zusätzlich aufgenommen worden. Dabei werden an mehreren Stellen auch noch entsprechenden Erläuterungen zum CAS Maple und zu ausgewählten Kommandos gemacht.

Der Leser findet dieses Preprint als Postscript-File *maple3.ps* sowie die Maple *mws*-Arbeitsblätter *kondit1.mws, kreis1.mws, geom1.mws, weihn1.mws, abl1.mws* auf der persönlichen Homepage im Internet.

Homepage
http://www.mathematik.tu-ilmenau.de/~neundorf/index_de.html

Navigator → Publications → Computeralgebra → MAPLE3

Kapitel 2

Fehlerquellen in Lösungsmethoden

Im Rahmen der Analyse des Problems als auch der Methode zu seiner Lösung ist es notwendig, Fehler abzuschätzen und in Abhängigkeit von den verfügbaren Rechenhilfsmitteln den Rechenaufwand zu bestimmen. Dies unterstützt die sachgerechte Entscheidung über die Verfahrensauswahl.
Hier soll zunächst eine kurze Übersicht mit Beispielen über die Arten von Fehlern gegeben werden.

2.1 Verfahrensfehler

Diskretisierungsfehler (vgl. [34], [35]).
- Ersetzen einer Funktion durch endlich viele Zahlen,
  numerische Integration auf der Basis der Funktionswerte $f_0, f_1, ..., f_n$,
  Polynominterpolation mit den Koeffizienten $a_0, a_1, ..., a_n$.
- Approximation einer Ableitung durch einen Differenzenquotienten.

Abbruchfehler, Iterationsfehler.
- Ersetzen eines infiniten Prozesses durch eine finites Verfahren.
- Partialsumme einer unendlichen Reihe.
- Grenzwert einer Iterationsfolge näherungsweise nach endlich vielen Schritten.

2.2 Eingangsfehler

Fehler treten in den Eingangsdaten als auch bei Rundung von Eingabedaten in maschinenkonforme Zahlen auf. Beide sind vom Charakter her analog zu behandeln.

2.3 Rundungsfehler und Kondition

Rundungsfehler können bei allen Rechenoperationen entstehen, sofern mit fester endlicher Stellenzahl und irgendeiner Rundung gerechnet wird.
Das Verhalten gegenüber Rundungsfehlern (und damit auch Eingangsfehlern) ist ein wesentliches Charakteristikum eines numerischen Algorithmus. Man studiert die Fehlerfortpflanzung von Störungen auf Zwischenergebnisse und das Schlussresultat, wie sich also Fehler vergrößern.
Die Kondition eines Problems ist der im ungünstigsten Falle auftretende Vergrößerungsfaktor für den Einfluss von relativen Eingangsfehlern auf relative Resultatsfehler. Unvermeidbare Eingangsfehler durch Störungen und Rundungen bewegen sich entweder in abschätzbaren beschränkten Größenordnungen oder sie wachsen über Maßen an, je nachdem, ob das Problem gut oder schlecht konditioniert ist.
Analog ist die Kondition des mathematischen Ersatzproblems sowie des Näherungsverfahrens zu sehen (z. B. Differentialgleichung, Differenzenschema, Gleichungssystemlöser).

Man bemerke also:

- Bei gegebener Aufgabe (mit eindeutiger Lösung) können zwei verschiedene Berechnungsverfahren völlig andere Ergebnisse liefern, weil sie unterschiedliches Fehlerverhalten haben.
- Die gute Kondition einer Aufgabe sollte möglichst in das entsprechende Ersatzproblem “hinübergerettet“ werden. Die Ersetzung einer schlecht gestellten Aufgabe durch ein “gut gestelltes“ Ersatzproblem heißt Regularisierung. Dabei muss man einen zusätzlichen Verfahrensfehler in Kauf nehmen.
- Wenn für ein Ersatzproblem mit gegebener (guter) Kondition das Verfahren eine wesentliche Verstärkung der Fehlerfortpflanzung bewirkt, ist das Verfahren schlecht konditioniert.
- Computerrealisierungen in Gleitpunktarithmetik (GP-Arithmetik) erzeugen notwendigerweise Rundungsfehler.
- Wichtig ist die Herausarbeitung von Begriffen und Prinzipien der Fehleranalyse und die Untersuchung des Fehlerverhaltens beginnend bei einfachen Basisalgorithmen der linearen Algebra (vgl. [29]).
- Erstellung einer Klassifikation von gut konditionierten Lösungsverfahren in den verschiedenen Teilgebieten der Numerischen Analysis.
  Die numerische Stabilität ist eine Mindestforderung an ein vernünftiges Verfahren. Liegt sie nicht vor, so können für gewisse Probleme beliebig große Genauigkeitsverluste in Bezug auf das unvermeidliche Fehlerniveau auftreten. Die bestmögliche Qualität eines Algorithmus stellt die numerische Gutartigkeit dar (vgl. [29]).

Kapitel 3

Anwendungen und Demonstrationen

Die hier durchgeführten Überlegungen sollen nunmehr an einigen ausgewählten Beispielen demonstriert werden. Sie beinhalten Anwendungsfälle, Probleme, Begriffe, Eigenschaften, Darstellungen oder Verfahren wie

- Lösung von linearen Gleichungssystemen,
- Matrixeigenschaften, wie Norm, Kondition, Eigenwerte,
- korrekt gestellte Aufgaben,
- Subtraktionskatastrophe,
- Termumformungen, mathematische und numerische Äquivalenz,
- Fehlerfortpflanzung und Fehleranalyse,
- Stabilität,
- Näherungszahlen und Stellenwertsysteme,
- interne Zahlendarstellung und Zahlenformate,
- Kurvendiskussion,
- rekursive und iterative Algorithmen,
- Verarbeitung von umfangreichen Datenmengen,
- spezielle Computereffekte,
- geometrische Sachverhalte,
- grafische Animationen.

Zahlreiche Sachverhalte sind dem Preprint M09/95 [37] entnommen.

Die Beispiele weisen darauf hin, wie komplex die Verwendung des Computers als Werkzeug und Medium der Lernens und Lehrens allgemein und besonders in der Mathematikausbildung zu sehen ist.
Der Komfort des Werkzeugs Computer einschließlich der Software kann sich aber nur “entfalten“ durch den an Ideen reichen und kreativen Nutzer in der Einheit mit seiner gezielten Nutzung, gründlichen Analyse und ständigen Weiterentwicklung.

3.1 Lösung eines linearen Gleichungssystems und Matrixkondition

Gesucht sei die Lösung des linearen Gleichungssystems

$$Ax = b \tag{3.1}$$

mit $x, b \in \mathbb{R}^n$ und der regulären Matrix $A \in \mathbb{R}^{n,n}$ der Form

$$\left(\begin{array}{cc|c} 0.780 & 0.563 & 0.217 \\ 0.913 & 0.659 & 0.254 \end{array}\right).$$

Um einfach festzustellen, ob ein Vektor x Lösung des Systems ist, prüft man, ob das Residuum $r = b - Ax$ einen Nullvektor liefert. Nehmen wir also zwei Kandidaten für die Lösung und zwar

$$\begin{aligned} \bar{x} &= (0.341,\ -0.087)^T, \\ \hat{x} &= (0.999,\ -1.001)^T, \end{aligned}$$

und berechnen dafür das Residuum. Es gilt entsprechend

$$\begin{aligned} \bar{r} &= (0.000001,\ 0)^T, \\ \hat{r} &= (0.001343,\ 0.001572)^T. \end{aligned}$$

Die "Güte" des Fehlers könnte den Betrachter dazu verleiten, $\bar{x}$ als den besseren Vorschlag zu akzeptieren. Das ist jedoch ein Trugschluss bei Kenntnis der exakten Lösung $x^* = (1, -1)^T$. Der Grund ist, dass die Matrix A eine schlechte Kondition hat. Kennzeichen dafür sind unter anderem:

- Die Matrix A ist fast singulär.
- Die Determinante von A ist nahe Null, denn $\det(A) = 10^{-6}$.
- Wenn die Matrix A Elemente der Größenordnung $\mathcal{O}(1)$ besitzt, dann hat die inverse Matrix A^{-1} betragsmäßig große Elemente, hier

$$A^{-1} = \begin{pmatrix} 659000 & -563000 \\ -913000 & 780000 \end{pmatrix}, \quad \det(A^{-1}) = 10^6.$$

- Das Spektrum der Eigenwerte von A ist sehr "breit". Es liegen Größenordnungen zwischen dem betragsmäßig kleinsten ($\neq 0$) und größten Eigenwert, denn sie betragen hier

$$\begin{aligned} \lambda_1 &= 0.000\,000\,694\,927\,37, \\ \lambda_2 &= 1.438\,999\,305\,072\,63. \end{aligned}$$

- Eine einfache geometrische Interpretation ist die Charakterisierung der Lösung als Schnittpunkt zweier Geraden, die sich hier in einem extrem spitzen Winkel schneiden.

Mit der Zeilensummennorm

$$\|A\|_\infty = \max_{1\le i\le n} \sum_{j=1}^{n} |a_{ij}| \tag{3.2}$$

beträgt die Kondition

$$\mathrm{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty \approx 2.5 \cdot 10^6. \tag{3.3}$$

Der Exponent $t = 6$ im Konditionswert charakterisiert im Groben die Reduzierung der gültigen Mantissenstellen der in Rechnungen benutzten GP-Arithmetik.

Rechnungen in Maple

```
> restart: with(plots): with(plottools): with(linalg):
```

Definition des LGS mit Matrix A (Float-Zahlen bzw. symbolisch)

```
> Digits:=10:
  A:=matrix(2,2,[0.780,0.563,0.913,0.659]);
  AA:=matrix(2,2,(i,j)->convert(A[i,j],rational));
  rank(A);
  det(A);
  norm(A);
  inverse(A);
  inverse(AA);
  b:=vector([0.217,0.254]);
  xs:=vector([1,-1]);
 `Ax*-b`=evalm(A&*xs-b);
```

$$A := \begin{bmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{bmatrix}$$

$$AA := \begin{bmatrix} \frac{39}{50} & \frac{563}{1000} \\ \frac{913}{1000} & \frac{659}{1000} \end{bmatrix}$$

$$2$$

$$0.1\,10^{-5}$$

$$1.572$$

$$\begin{bmatrix} 659000.0000 & -563000.0000 \\ -913000.0000 & 780000.0000 \end{bmatrix}$$

$$\begin{bmatrix} 659000 & -563000 \\ -913000 & 780000 \end{bmatrix}$$

$$b := [0.217,\ 0.254]$$

$$xs := [1,\ -1]$$

$$Ax^* - b = [0.,\ 0.]$$

Rechnung mit schwacher Gleitpunktarithmetik

```
> Digits:=5;
  evalm(A);
  rank(A);
  det(A);
  norm(A);
  inverse(A);
```

$$Digits := 5$$

$$\begin{bmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{bmatrix}$$

$$1$$

$$0.$$

$$1.572$$

```
Error, (in inverse) singular matrix
```

Test von zwei Loesungskandidaten und Berechnung der Residuen dazu

```
> Digits:=10;
  xq := vector([0.341,-0.087]);
  xd := vector([0.999,-1.001]);
  multiply(A,xq);
  rq := evalm(b-multiply(A,xq));
  rd := evalm(b-A&*xd);
```

$$Digits := 10$$

$$xq := [0.341,\ -0.087]$$

$$xd := [0.999,\ -1.001]$$

$$[0.216999,\ 0.254000]$$

$$rq := [0.1\,10^{-5},\ 0.]$$

$$rd := [0.001343,\ 0.001572]$$

Loesung des LGS mit Gleitpunktarithmetik bei unterschiedlicher Mantissenlaenge

```
> Digits:=16;
  erg:=linsolve(A,b);
  erg[1];
  erg[2];
  i:='i':
  printf(`Digits         x[1]                        x[2]\\n`):
  for i from 5 to 16 do
   Digits:=i:
   erg:=linsolve(A,b):
   if rank(A)<2 then
    printf(`%2d       Matrix A singulaer\n`,i)
   else
    printf(`%2d       %.16e  %.16e\n`,i,erg[1],erg[2])
   end if:
  end do:
```

$$Digits := 16$$
$$erg := [1.000000000000000, -1.000000000000000]$$
$$1.000000000000000$$
$$-1.000000000000000$$

```
Digits         x[1]                        x[2]
 5       Matrix A singulaer
 6       Matrix A singulaer
 7       Matrix A singulaer
 8       9.7422161000000000e-01  -9.6428571000000000e-01
 9       1.0006597800000000e+00  -1.0009140800000000e+00
10       9.9993410670000000e-01  -9.9990870920000000e-01
11       1.0000131801000000e+00  -1.0000182602000000e+00
12       9.9999802300600000e-01  -9.9999726100400000e-01
13       9.9999986820020000e-01  -9.9999981740000000e-01
14       9.9999998023004000e-01  -9.9999997261000000e-01
15       9.9999999934100200e-01  -9.9999999908700000e-01
16       1.0000000000000000e+00  -1.0000000000000000e+00
```

3.1.1 Matrizen und Eigenschaften

Explizite oder direkte Gleichungssystemlöser müssen numerisch stabil, wenn nicht gutartig sein, um bei schlecht konditionierter Systemmatrix noch vertretbare Lösungen zu erzeugen.
Aber schon für solche Matrizen, wie die symmetrische und positiv definite (spd) **Hilbert-Matrix**

$$A = (a_{ij}), \quad a_{ij} = \frac{1}{i+j-1}, \tag{3.4}$$

deren Inverse $A^{-1} = (\alpha_{ij})$ die ganzzahligen Elemente

$$\alpha_{ij} = \frac{(-1)^{i+j}}{i+j-1}\gamma_i\gamma_j, \ \gamma_i = \frac{(n+i-1)!}{(i-1)!^2(n-i)!}, \quad i,j = 1,2,...,n,$$

enthält, oder der ganzzahligen **Boothroyd/Dekker-Matrix**

$$A = (a_{ij}), \quad a_{ij} = \binom{n+i-1}{i-1}\binom{n-1}{n-j}\frac{n}{i+j-1}, \tag{3.5}$$

mit der Inversen $A^{-1} = (\alpha_{ij})$ und ihren Elementen

$$\alpha_{ij} = (-1)^{i+j}a_{ij}, \quad i,j = 1,2,...,n,$$

sind wegen ihrer schlechten Kondition für die Lösung eines Systems mit gegebener GP-Arithmetik bei nicht allzugroßer Dimension Grenzen gesetzt.

Rechnungen in Maple

Hilbert-Matrix A spd

```
> Digits:=16:
  i:='i': j:='j':
  n:=5:
  A:=matrix(n,n):
  A:=matrix(n,n,(i,j)->1/(i+j-1)):   # hilbert(n)
  `A`=evalm(A);
  `rank(A)`=rank(A);
  `det(A)`=det(A);
  `inv(A)`=inverse(A);

  # Normen
  `norm(A,2)`=norm(A,2);                    # Spektralnorm
  evalf(%);
  `norm(A,1)`=norm(A,1);                    # Spaltensummennorm
  evalf(%);
  `norm(A)=norm(A,infinity)`=norm(A);       # Zeilensummennorm
  evalf(%);
  `norm(A,frobenius)`=norm(A,frobenius);    # Frobenius-Norm
  evalf(%);
  # Konditionen dazu, auch Kommando cond(A,*)
  `cond(A,2)`=evalf(norm(A,2)*norm(inverse(A),2));
  `cond(A,1)`=evalf(norm(A,1)*norm(inverse(A),1));
  `cond(A)=cond(A,infinity)`=evalf(norm(A)*norm(inverse(A)));
  `cond(A,frobenius)`=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
```

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}$$

$$rank(A) = 5$$

$$det(A) = \frac{1}{266716800000}$$

$$inv(A) = \begin{bmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{bmatrix}$$

$$norm(A,2) =$$

$$\frac{1}{5} RootOf(-1 + 3700542505\,\_Z - 1582832489513760\,\_Z^2 + 487666609069973760\,\_Z^3$$
$$-455148325561466880\,\_Z^4 + 7284515983589376\,\_Z^5,\ index = 5)^{1/2}$$

$$norm(A,2) = 1.567050691098231$$
$$norm(A,1) = \frac{137}{60}$$
$$norm(A,1) = 2.283333333333333$$
$$norm(A) = norm(A,infinity) = \frac{137}{60}$$
$$norm(A) = norm(A,infinity) = 2.283333333333333$$
$$norm(A,frobenius) = \frac{\sqrt{15871330}}{2520}$$
$$norm(A,frobenius) = 1.580906263272022$$
$$cond(A,2) = 476607.2502425608$$
$$cond(A,1) = 943656.$$
$$cond(A) = cond(A,infinity) = 943656.$$
$$cond(A,frobenius) = 480849.1169947188$$

EW

- **Eigenvals(A)** returns an array of the eigenvalues of A. The function Eigenvals itself is inert. To actually compute the eigenvalues and eigenvectors, the user must evaluate the inert function in the floating point domain, by **evalf(Eigenvals(A))**.
- The call **eigenvalues(A)** returns for a symbolic case a sequence of the eigenvalues of A computed by solving the characteristic polynomial $\det(\lambda I - A) = 0$ or for larger dimension (greater than four) the eigenvalues are expressed using Maple's **RootOf** notation for algebraic extensions. If A contains floating-point numbers, a numerical method is used where all arithmetic is done at the precision specified by **Digits**.

```
> charpoly(A,lambda);
  Eigenvals(A);
  evalf(Eigenvals(A));
  eigenvals(A);
  evalf(eigenvals(A));
```

$$\lambda^5 - \frac{563}{315}\lambda^4 + \frac{735781}{2116800}\lambda^3 - \frac{852401}{222264000}\lambda^2 + \frac{61501}{53343360000}\lambda - \frac{1}{266716800000}$$

$$\text{Eigenvals}\left(\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}\right)$$

$$[0.3287928771915000\,10^{-5},\ 0.0003058980401511738,\ 0.01140749162341977,\ 0.2085342186110133,\ 1.567050691098231]$$

$$\%1 := -1+61501\,\_Z-40915248\,\_Z^2+741667248\,\_Z^3-762725376\,\_Z^4+85349376\,\_Z^5$$

$$\frac{1}{5}\,RootOf(\%1,\ index=1),\ \frac{1}{5}\,RootOf(\%1,\ index=2),\ \frac{1}{5}\,RootOf(\%1,\ index=3),\ \frac{1}{5}\,RootOf(\%1,\ index=4),\ \frac{1}{5}\,RootOf(\%1,\ index=5)$$

$$0.3287928772171862\,10^{-5},\ 0.0003058980401511918,\ 0.01140749162341981,\ 0.2085342186110134,\ 1.567050691098231$$

Prozedur fuer die Berechnung der Spektralnorm ueber EW von AA^T

```
> norm2:=proc(A::matrix)
        local n,i,B,EVB,seq_EVB;
   B:=evalm(A&*transpose(A));
   n:=linalg[rowdim](B);
   EVB:=evalf(Eigenvals(B));
   seq_EVB:=seq(EVB[i],i=1..n);
   sqrt(max(seq_EVB));
  end:
> norm2(A);
  evalf(norm(A,2));
  Digits:=10:
```

$$1.567050691098231$$
$$1.567050691098231$$

Kondition der Hilbert-Matrix abhaengig von der Dimension n

```
> Digits:=16:
  i:='i': j:='j':
  printf(` n          cond(H(n,n))\n`):
  for n from 1 to 10 do
   A:=matrix(n,n,(i,j)->1/(i+j-1)):
   erg:=evalf(norm(A,2)*norm(inverse(A),2));
   printf(`%2d    %.16e  \n`,n,erg);
  end do:
```

```
n          cond(H(n,n))
 1    1.0000000000000000e+00
 2    1.9281470067903970e+01
 3    5.2405677758606080e+02
 4    1.5513738738932590e+04
 5    4.7660725024256080e+05
 6    1.4951058640131220e+07
 7    4.7536735498817890e+08
 8    1.5257575741646940e+10
 9    4.9315492697154210e+11
10    1.6026286870216880e+13
```

Boothroyd/Dekker-Matrix A

```
> Digits:=16:
  i:='i': j:='j':
  n:=5:
  A:=matrix(n,n):
  for i from 1 to n do
   for j from 1 to n do
    A[i,j]:=binomial(n+i-1,i-1)*binomial(n-1,n-j)*n/(i+j-1)
   end do:
  end do:
  `A`=evalm(A);
  `rank(A)`=rank(A);
  `det(A)`=det(A);
  `inv(A)`=inverse(A);

  # Normen
  `norm(A,2)`=norm(A,2);                          # Spektralnorm
  evalf(%);
  `norm(A,1)`=norm(A,1);                          # Spaltensummennorm
  `norm(A)=norm(A,infinity)`=norm(A);             # Zeilensummennorm
  `norm(A,frobenius)`=norm(A,frobenius);          # Frobenius-Norm
  evalf(%);

  # Konditionen dazu
  `cond(A,2)`=evalf(norm(A,2)*norm(inverse(A),2));
  `cond(A,1)`=evalf(norm(A,1)*norm(inverse(A),1));
  `cond(A)=cond(A,infinity)`=evalf(norm(A)*norm(inverse(A)));
  `cond(A,frobenius)`=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
  Digits:=10:
```

$$A = \begin{bmatrix} 5 & 10 & 10 & 5 & 1 \\ 15 & 40 & 45 & 24 & 5 \\ 35 & 105 & 126 & 70 & 15 \\ 70 & 224 & 280 & 160 & 35 \\ 126 & 420 & 540 & 315 & 70 \end{bmatrix}$$

$$rank(A) = 5$$

$$det(A) = 1$$

$$inv(A) = \begin{bmatrix} 5 & -10 & 10 & -5 & 1 \\ -15 & 40 & -45 & 24 & -5 \\ 35 & -105 & 126 & -70 & 15 \\ -70 & 224 & -280 & 160 & -35 \\ 126 & -420 & 540 & -315 & 70 \end{bmatrix}$$

$$norm(A,2) = \sqrt{RootOf(\_Z^4 - 786254\,\_Z^3 + 132041256\,\_Z^2 - 786254\,\_Z + 1,\ index = 4)}$$

$$norm(A,2) = 886.6149259774036$$

$$norm(A,1) = 1001$$

$$norm(A) = norm(A, infinity) = 1471$$

$$norm(A, frobenius) = \sqrt{786255}$$
$$norm(A, frobenius) = 886.7102119632998$$
$$cond(A, 2) = 786086.0269659169$$
$$cond(A, 1) = 0.1002001\,10^7$$
$$cond(A) = cond(A, infinity) = 0.2163841\,10^7$$
$$cond(A, frobenius) = 786255.$$

Dem Leser überlassen wir ähnliche Untersuchungen zur **Zielke-Matrix**.

Dabei handelt es sich um symmetrische, indefinite und parameterabhängige Matrizen $A = (a_{ij})$ mit $A^{-1} = (\alpha_{ij})$, wobei

$$a_{ij} = \begin{cases} \alpha + 1 & \text{für } i = 1(1)n,\ j = 1(1)n - i, \\ \alpha - 1 & \text{für } i = j = n, \\ \alpha & \text{sonst,} \end{cases} \qquad \alpha \in \mathbb{R}, \tag{3.6}$$

$$\alpha_{ij} = \begin{cases} -\alpha & \text{für } i = j = 1, \\ \alpha & \text{für } i = 1, j = n \text{ und } i = n, j = 1, \\ -\alpha - 1 & \text{für } i = j = n, \\ 1 & \text{für } i = 1(1)n - 1,\ j = n - i, \\ -1 & \text{für } i = 2(1)n - 1,\ j = n - i + 1, \\ 0 & \text{sonst,} \end{cases} \tag{3.7}$$

EW: $|\lambda_{max}| \approx n\alpha,\ |\lambda_{min}| \approx 1/(2\alpha)$,

Konditionszahlen: $\mathrm{cond}_F(A) = \mathrm{cond}_2(A) \approx 2n\alpha^2$,

$n = 5$:

$$A = \begin{pmatrix} \alpha{+}1 & \alpha{+}1 & \alpha{+}1 & \alpha{+}1 & \alpha \\ \alpha{+}1 & \alpha{+}1 & \alpha{+}1 & \alpha & \alpha \\ \alpha{+}1 & \alpha{+}1 & \alpha & \alpha & \alpha \\ \alpha{+}1 & \alpha & \alpha & \alpha & \alpha \\ \alpha & \alpha & \alpha & \alpha & \alpha{-}1 \end{pmatrix}, \quad A^{-1} = \begin{pmatrix} -\alpha & 0 & 0 & 1 & \alpha \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 \\ \alpha & 0 & 0 & 0 & -\alpha{-}1 \end{pmatrix}.$$

Zum Schluss betrachten wir zur **Hilbert-Matrix** die Durchführung des Gaußschen Eliminationverfahrens mit Spaltenpivotisierung für die Ermittlung der Inversen und die Berechnung der Determinante $\det(A)$ bei wachsender Dimension n. Dabei tendieren die sogenannten Pivotelemente gegen Null.
Zu Grunde liegen Implementierungen in TP mit den GP-Format *double* (64 Binärstellen, 15-16 Dezimalstellen der Mantisse) und *extended* (80 Binärstellen, 19-20 Dezimalstellen der Mantisse) sowie Rechnungen in Matlab (*double* Präzision) und Maple.

Generell wird sich bei numerischen Rechnungen die Anzahl der signifikanten Stellen im Ergebnis proportional zur wachsenden Dimension n verringern. So ist z. B. bei der Berechnung von $\det(A)$ mit TP *double* Präzision bei $n = 13$ höchsten noch die Größenordnung des Wertes verlässlich, bei $n = 16$ nicht einmal diese, was der Vergleich mit der exakten Auswertung im CAS Maple zeigt.

Programm	n	7	10	13	16	19
TP	$\det(A)$	4.8E-25	2.2E-53	2.6E-92	1.8E-135	-3.9E-180
double	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.8E+8	1.2E+13	2.3E+17	5.7E+17	5.9E+17
TP	$\det(A)$	4.8E-25	2.2E-53	1.4E-92	-3.2E-141	1.1E-192
extended	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.8E+8	1.2E+13	4.2E+17	7.3E+20	7.5E+20
Matlab	$\det(A)$	4.8E-25	2.2E-53	1.4E-92	2.4E-135	-2.2E-180
double	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.8E+8	1.2E+13	1.1E+17 (a)	6.8E+17	1.9E+18
				1.3E+17 (b)		

Tab. 3.1 Berechnungen für die Hilbert-Matrix mit TP und Matlab, wobei
(a): $A\backslash I$,
(b): $inv(A)$ (ab $n \geq 13$ mit Warnungen)

Programm	n	7	10	13	16	19
Maple	$\det(A)$	4.8E-25	2.2E-53	1.4E-92	1.4E-142	2.0E-203
exakt	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.8E+8	1.2E+13	4.2E+17	1.5E+22	5.4E+26

Tab. 3.2 Berechnungen für die Hilbert-Matrix mit Maple

Für die Dimensionen $n = 16$ und 19 sind die Berechnungen der Determinante und Kondition der Matrix A in TP bzw. Matlab mit der vorliegenden GP-Arithmetik nicht vertretbar.

3.2 Iterationsverfahren für lineare Gleichungssysteme

Iterationsverfahren (IV) für das Gleichungssystem $Ax = b$ sind, wenn sie konvergieren, selbstkorrigierende Methoden. Sie können mit einem angenommenen Startvektor sofort bzw. im Nachgang an ein direktes Verfahren eingesetzt werden. In jedem Fall erhoffen wir uns eine Korrektur der Näherungslösung. Deshalb spricht man auch von Defekt- oder Residuenkorrekturverfahren oder einfach Residueniteration.
Sei A regulär mit $a_{ii} > 0$ und der Zeilensummennorm $\|A\|_\infty = 1$.
Dies ist erreichbar durch entsprechende Zeilenvertauschung von A und eine sinnvolle Skalierung (Normalisierung, Äquilibrierung) gemäß

$$\begin{aligned} A' &= DA, \quad A' = (a'_{ij}), \\ D &= \operatorname{diag}(d_1, d_2, ..., d_n), \\ d_i &= \operatorname{sign}(a_{ii}) \Big(\sum_{j=1}^{n} |a_{ij}| \Big)^{-1}, \quad i = 1, 2, ..., n. \end{aligned} \tag{3.8}$$

Damit haben wir mit der vorgeschlagenen Zeilenskalierung auch eine erste Variante der sogenannten Vorkonditionierung der Matrix durchgeführt und erhalten

$$\sum_{j=1}^{n} |a'_{ij}| = 1 \quad \text{und} \quad a'_{ii} \geq 0, \quad i = 1, 2, ..., n.$$

Die Kondition von A, ausgedrückt mit der Zeilensummennorm, wird sich dabei nicht verschlechtern, oftmals jedoch deutlich besser werden. Wenn wir die Vorzeichenbedingung fallen lassen, ist die Skalierung weit weniger effizient.
Das Problem formt man nun häufig um in eine Fixpunktgleichung der Form

$$x = x + (b - Ax), \quad r = b - Ax \text{ Residuum}, \tag{3.9}$$

und zur numerischen Implementierung in die Iterationsgestalt (Residualform)

$$x^{(m+1)} = x^{(m)} + (b - Ax^{(m)}) = (I - A)x^{(m)} + b. \tag{3.10}$$

Dieses "naive" IV mit der Iterationsmatrix $H = I - A$ konvergiert nicht, wenn der Spektralradius $\rho(H) > 1$ ist, und kann auch leicht bei diagonaldominanter Matrix A versagen. Somit wird einfach auf die modifizierte Form der Fixpunktpunktgleichung

$$x = x + \omega(b - Ax), \ \omega > 0 \ \text{ Parameter}, \tag{3.11}$$

zurückgegriffen, die nunmehr schon eine ganze Reihe von Vorzügen besitzt.
Das entsprechende IV wird als
- Richardson-Iteration,

- Relaxation,
- ω-Jacobi-Verfahren oder
- Jacobi Overrelaxation (JOR)

bezeichnet.

Wenn man noch einen Schritt weiter geht zu

$$\begin{array}{rcl} x & = & x + W^{-1}(b - Ax), \quad W \in \mathbb{R}^{n,n} \text{ regulär}, \\ x & = & x + \omega W^{-1}(b - Ax), \end{array} \tag{3.12}$$

ist man bei der Klasse der semiterativen Verfahren, der beiden Basis-IV von Jacobi (Gesamtschrittverfahren, GSV) und Gauß-Seidel (Einzelschrittverfahren, ESV), der Newton-Verfahren, Gradientenverfahren mit/ohne Vorkonditionierung unter Ausnutzung von Zerlegungen (Splitting) der Matrix A für die Wahl der Skalierungsmatrix (Vorkonditionierungsmatrix, Wichtung) W.

Dazu nehmen wir z. B. eine einfache Zerlegung der Matrix A wie folgt vor:

$$A \;=\; D - E - F \;=\; D(I - L - U) \tag{3.13}$$

mit

I	Einheitsmatrix,
D	Diagonalmatrix, $D = \text{diag}(A) = \text{diag}(a_{11}, a_{22}, ..., a_{nn})$,
L	linke Dreiecksmatrix (auch Diagonale ist Null), $L = (l_{ij})$, $E = DL$,
U	rechte Dreiecksmatrix (auch Diagonale ist Null), $U = (u_{ij})$, $F = DU$.

Für ein Gleichungssystem mit positiver und diagonaldominanter Matrix können wir die konvergente Richardson-Iteration

$$\begin{array}{rcl} x^{(m+1)} & = & x^{(m)} + \dfrac{1}{\|A\|_\infty}(b - Ax^{(m)}), \quad m = 0, 1, ..., \\ & & x^{(0)} \;\; \text{beliebiger Startvektor}, \end{array} \tag{3.14}$$

anwenden.

Die Matrix-Vektor-Form des GSV ist

$$\begin{array}{rcl} x^{(m+1)} & = & x^{(m)} + D^{-1}r^{(m)} \;=\; Jx^{(m)} + D^{-1}b, \\ & & J \;=\; I - D^{-1}A \;\; \text{Iterationsmatrix}, \end{array} \tag{3.15}$$

die des ESV

$$\begin{array}{rcl} x^{(m+1)} & = & x^{(m)} + (D - E)^{-1}r^{(m)} \;=\; H_1x^{(m)} + [D(I - L)]^{-1}b, \\ & & H_1 \;=\; I - [D(I - L)]^{-1}A \;\; \text{Iterationsmatrix}. \end{array} \tag{3.16}$$

Wir formulieren ein hinreichendes Konvergenzkriterium für diese IV.

Sei A diagonaldominant. Dann konvergieren das GSV und ESV für beliebige Startvektoren. Mehr noch, es gilt die Ungleichung $\|H_1\|_\infty \leq \|J\|_\infty < 1$. Hinreichende und notwendige Bedingung für die Konvergenz der IV werden mittels dem Spektralradius formuliert. Wenn der Spektralradius der Iterationsmatrix kleiner als 1 ist, dann

haben wir nicht nur die gesicherte Konvergenz des IV, sondern irgendwelche Fehler im Lösungsprozess - auch künstlich eingebrachte Störungen - werden stets gedämpft. Darüber hinaus sind garantierte Abschätzungen der Konvergenzrate und der Fehler möglich.

Bemerkung 3.1 Robustheit (Stabilität, Konvergenz) und Effizienz (Zeitaufwand, Speicherbedarf) sind zwei Seiten eines Gleichungssystemlösers. Gerade für großdimensionierte Systeme leben diese Eigenschaften "in unterschiedlichen Welten" und erfordern einen Kompromiss. Sogenannte Polyalgorithmen, die zwischen verschiedenen Verfahren geeignet und geschickt umschalten, sind ein möglicher Ausweg, verlangen aber auch im Umgang einen reichen Erfahrungsschatz.
Bei iterativen Methoden spielen der Spektralradius der Iterationsmatrix sowie die schnelle Matrix-Vektor-Multiplikation eine große Rolle.

Beispiel 3.1 $Ax = b$ mit

$$A = \begin{pmatrix} 12 & -2 & 3 \\ -1 & 8 & -2 \\ -1 & 3 & 12 \end{pmatrix}, \quad b = (18, -32, 12)^T.$$

Die exakte Lösung ist

$$x^* = \Big(\frac{540}{1211}, -\frac{600}{173}, \frac{2306}{1211}\Big)^T = (0.4459124690, -3.468208092, 1.904211396)^T.$$

Die Matrix A ist diagonaldominant und hat positive Diagonalelemente.
Somit können die Fixpunktgleichungen $x = Hx + c$ mit

$$\begin{aligned} H &= H_R = I - \frac{1}{\|A\|_\infty} A, \quad c = \frac{1}{\|A\|_\infty} b, \\ H &= J = I - D^{-1}A, \quad c = D^{-1}b, \\ H &= H_1 = I - [D(I-L)]^{-1}A, \quad c = [D(I-L)]^{-1}b, \end{aligned}$$

die zum Richarson-IV, GSV bzw. ESV führen, zur Iteration verwendet werden kann.

Die IV $x^{(m+1)} = Hx^{(m)} + c$ benutzen als Startvektor den Nullvektor und werden abgebrochen, wenn $\|x^{(m)} - x^{(m-1)}\|_2 \leq \varepsilon,\ \varepsilon > 0$, oder die vorgegeben maximale Iterationsanzahl erreicht werden.

Man kann auch den Spektralradius $\rho(H)$ der Iterationsmatrix sowie wegen der Kenntnis von x^* den Fehlerverlauf $\|x^{(m)} - x^*\|_2$ kontrollieren.

Konvergenzbetrachtungen zeigen die wachsende Konvergenzrate der IV

$$R(H) = -\ln(\rho(H)) > 0 \tag{3.17}$$

in der genannten Reihenfolge, die mit fallendem Spektralradius der Iterationsmatrizen wächst.

Rechnungen in Maple

LGS $Ax = b$, A streng diagonaldominant, $\mathrm{diag}(A) > 0$

```
> n:=3:
  A:=matrix(n,n,[[12,-2,3],[-1,8,-2],[-1,3,12]]);
  b:=vector(n,[18,-32,12]);
```

$$A := \begin{bmatrix} 12 & -2 & 3 \\ -1 & 8 & -2 \\ -1 & 3 & 12 \end{bmatrix}$$

$$b := [18, -32, 12]$$

Eigenschaften der Matrix und Loesung des LGS

```
> rank(A);
  det(A);
  norm(A);              # Zeilensummennorm  norm(A,infinity)
  evalm(inverse(A)&*b);
  xs:=linsolve(A,b);
  evalf(evalm(xs));
  lambda:=evalf(eigenvals(A));
  alambda:=seq(abs(lambda[i]),i=1..n);
  rho:=max(alambda);   # Spektralradius von A
```

$$3$$

$$1211$$

$$17$$

$$\left[\frac{540}{1211}, \frac{-600}{173}, \frac{2306}{1211}\right]$$

$$xs := \left[\frac{540}{1211}, \frac{-600}{173}, \frac{2306}{1211}\right]$$

$$[0.4459124690, -3.468208092, 1.904211396]$$

$$\lambda := 8.136836128, 11.93158194 - 2.542970428\,I, 11.93158194 + 2.542970428\,I$$

$$alambda := 8.136836128, 12.19956336, 12.19956336$$

$$\rho := 12.19956336$$

Matrixzerlegung $A = D - E - F = D(I - L - U)$

```
> evalm(A);
  DD:=diag(seq(A[i,i],i=1..n));
      # DD:=matrix(n,n,(i,j)->if i=j then A[i,j] else 0 fi);
  L1:=Matrix(n,[A],shape=triangular[lower]);
  U1:=Matrix(n,[A],shape=triangular[upper]);
  E:=evalm(U1-A); # E:=matrix(n,n,(i,j)->if i>j then -A[i,j] else 0 fi);
  F:=evalm(L1-A); # F:=matrix(n,n,(i,j)->if i<j then -A[i,j] else 0 fi);
  evalm(DD-E-F);  # = A
```

```
L:=evalm(inverse(DD)&*E);
U:=evalm(inverse(DD)&*F);
II:=evalm(array(identity,1..n,1..n));
evalm(DD&*(II-L-U));     # = A
```

$$\begin{bmatrix} 12 & -2 & 3 \\ -1 & 8 & -2 \\ -1 & 3 & 12 \end{bmatrix}$$

$$DD := \begin{bmatrix} 12 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 12 \end{bmatrix}$$

$$L1 := \begin{bmatrix} 12 & 0 & 0 \\ -1 & 8 & 0 \\ -1 & 3 & 12 \end{bmatrix}$$

$$U1 := \begin{bmatrix} 12 & -2 & 3 \\ 0 & 8 & -2 \\ 0 & 0 & 12 \end{bmatrix}$$

$$E := \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & -3 & 0 \end{bmatrix}$$

$$F := \begin{bmatrix} 0 & 2 & -3 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 12 & -2 & 3 \\ -1 & 8 & -2 \\ -1 & 3 & 12 \end{bmatrix}$$

$$L := \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{8} & 0 & 0 \\ \frac{1}{12} & \frac{-1}{4} & 0 \end{bmatrix}$$

$$U := \begin{bmatrix} 0 & \frac{1}{6} & \frac{-1}{4} \\ 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 \end{bmatrix}$$

$$II := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 12 & -2 & 3 \\ -1 & 8 & -2 \\ -1 & 3 & 12 \end{bmatrix}$$

Prozedur
3 IV mit Variantenwahl RF(ω), GSV, ESV
mit zahlreichen Ausgaben und Dateiarbeit

```
> interface(verboseproc=0):    # keine Ausgabe der Prozedurdefinition

> IV:=proc(iv_lgs::name,datei::string,n::posint,A::matrix,b::vector,
           x0::vector,maxiter::posint,etol::numeric)

      local  m,err,II,DD,omega,H,c,x,xn,fh,fh1,form1,form2,form3,i,
             file1,lambda,alambda,rho;

      global xs,f_iv;

  file1:=fopen(datei,WRITE);
  fh1:=`%+13.9f`;       # Ausgabeformate einstellen
  fh:=fh1;
  for i from 2 to n do
   fh:=cat(fh,` `,fh1);
  end do;
  form1:=`%2d                          `||fh1||`     `||fh||`\n`;
  form2:=`%2d     `||fh1||`      `||fh1||`     `||fh||`\n`;
  form3:=`xs                                          `||fh||`\n`;

  x:=vector(n):
  xn:=vector(n):
  II:=diag(1$n);

  if iv_lgs=RF then
   omega:=1/norm(A,infinity);
   H:=evalm(II-omega*A);          # Iterationsmatrix
   c:=evalm(omega*b);
  elif iv_lgs=GSV then
   DD:=diag(seq(A[i,i],i=1..n));
   H:=evalm(II-inverse(DD)&*A);   # Iterationsmatrix
   c:=evalm(inverse(DD)&*b);
  elif iv_lgs=ESV then
   DD:=matrix(n,n,(i,j)->if i>=j then A[i,j] else 0 end if);
   H:=evalm(II-inverse(DD)&*A);   # Iterationsmatrix
   c:=evalm(inverse(DD)&*b);
  end if;

  # EW und Spektralradius der Iterationsmatrix
  lambda:=evalf(eigenvals(H));
  alambda:=seq(abs(lambda[i]),i=1..n);
  rho:=max(alambda);
  lprint(`Spektralradius  rho(H) = `,rho):

  x:=evalm(x0);
  m:=0;
  err:=1;
  f_iv[1]:=evalf(norm(x-xs,2));
```

```
fprintf(default,‘ \n‘);
fprintf(file1,  ‘ \n‘);
fprintf(default,‘Iterationsverlauf\n‘);
fprintf(file1,  ‘Iterationsverlauf\n‘);
fprintf(default,‘ m  ||x(m)-x(m-1)||_2   ||x(m)-xs||_2     x(m)\n‘);
fprintf(file1,  ‘ m  ||x(m)-x(m-1)||_2   ||x(m)-xs||_2     x(m)\n‘);
fprintf(default,form1,m,f_iv[1],seq(x[i],i=1..n));
fprintf(file1,  form1,m,f_iv[1],seq(x[i],i=1..n));

while (m<maxiter) and (err>etol)  do
 xn:=evalm(H&*x+c);
 m:=m+1;
 err:=evalf(norm(xn-x,2));
 f_iv[m+1]:=evalf(norm(xn-xs,2));
 x:=evalm(xn);
 fprintf(default,form2,m,err,f_iv[m+1],seq(x[i],i=1..n));
 fprintf(file1,  form2,m,err,f_iv[m+1],seq(x[i],i=1..n));
end do:

fprintf(default,form3,seq(xs[i],i=1..n));
fprintf(file1,  form3,seq(xs[i],i=1..n));
fprintf(default,‘ \n‘);
fprintf(file1,  ‘ \n‘);
fclose(file1);

[x,m];
end:
```

RF(ω)

```
> dd:="C:/D/Neundorf/Maple3/rfi1.res":
          # nicht ‘C:/D/Neundorf/Maple3/rfi1.res‘
  x0:=vector(n,[0$n]);
  etol:=1e-16;
  maxiter:=10;
  f_iv:=vector(maxiter+1,[0$(maxiter+1)]):

  lprint(‘RF‘):
  erg:=IV(RF,dd,n,A,b,x0,maxiter,etol):
  xiter_rfi:=evalf(evalm(erg[1]));
  iter_rfi:=erg[2];
  f_rfi:=evalm(f_iv):
```

$$x0 := [0,\ 0,\ 0]$$
$$etol := 0.1\,10^{-15}$$
$$maxiter := 10$$

```
RF
‘Spektralradius  rho(H) = ‘, .5213625809
```

```
Iterationsverlauf
 m  ||x(m)-x(m-1)||_2  ||x(m)-xs||_2                  x(m)
 0                     +3.981623582   +0.000000000  +0.000000000  +0.000000000
 1     +2.272142108     +2.080045437   +1.058823529  -1.882352941   +.705882353
 2     +1.043193776     +1.108900692   +1.024221453  -2.733564014  +1.307958478
 3      +.546351977      +.573377354    +.807653165  -3.115408101  +1.633217993
 4      +.285222357      +.289670708    +.641635038  -3.292034339  +1.783527496
 5      +.144472631      +.145451730    +.545501413  -3.377624648  +1.849139737
 6      +.072035279      +.073486997    +.495578739  -3.420872997  +1.877886708
 7      +.035975073      +.037535649    +.470734563  -3.443323813  +1.891037134
 8      +.018174849      +.019367448    +.458465517  -3.455123852  +1.897405392
 9      +.009305991      +.010062882    +.452344924  -3.461343433  +1.900639061
10      +.004815108      +.005248011    +.449242387  -3.464615756  +1.902327678
xs                                     +.445912469  -3.468208092  +1.904211396
```

$$xiter\_rfi := [0.4492423865,\ -3.464615756,\ 1.902327678]$$
$$iter\_rfi := 10$$

Grafik

```
> x:=[seq(k,k=0..iter_rfi)]:
  ylog:=[seq(log10(f_rfi[k]/f_rfi[1]),k=1..iter_rfi+1)]:
  if f_rfi[iter_rfi+1]=0 then
   ylog:=[op(1..iter_rfi,ylog),-16]:
  end if:
  prfi:=[seq([x[k],ylog[k]],k=1..iter_rfi+1)]:

> k:='k':
  prfi1:=plot(prfi,k=-0.2..iter_rfi+0.2,color=green,thickness=2):
  prfi2:=textplot([[5,-0.5,`log10(||f_rfi[k]||_2/||f_rfi[0]||_2)`]]):
  plots[display]([prfi1,prfi2],axes=boxed,
        title=`RF(om) - Logarithmus des relativen Fehlers`);
```

GSV

```
> dd:="C:/D/Neundorf/Maple3/gsv1.res":
  x0:=vector(n,[0$n]);
  etol:=1e-16;
  maxiter:=10;
  f_iv:=vector(maxiter+1,[0$(maxiter+1)]):

  lprint(`GSV`):
  erg:=IV(RF,dd,n,A,b,x0,maxiter,etol):
  xiter_gsv:=evalf(evalm(erg[1]));
  iter_gsv:=erg[2];
  f_gsv:=evalm(f_iv):
```

$$x0 := [0,\ 0,\ 0]$$
$$etol := 0.1\,10^{-15}$$
$$maxiter := 10$$

```
GSV
‘Spektralradius  rho(H) = ‘, .2856869311

Iterationsverlauf
 m  ||x(m)-x(m-1)||_2  ||x(m)-xs||_2                    x(m)
 0                     +3.981623582   +0.000000000  +0.000000000  +0.000000000
 1     +4.387482194     +1.487111766   +1.500000000  -4.000000000  +1.000000000
 2     +1.515687642      +.276628028    +.583333333  -3.562500000  +2.125000000
 3      +.325098105      +.107207345    +.375000000  -3.395833333  +1.939236111
 4      +.119358428      +.024229943    +.449218750  -3.468315972  +1.880208333
 5      +.025058925      +.008191774    +.451895255  -3.473795573  +1.904513889
 6      +.009617331      +.002298819    +.444905599  -3.467384621  +1.906106831
 7      +.002334621      +.000564253    +.445575855  -3.467860092  +1.903921622
 8      +.000680144      +.000208248    +.446042913  -3.468322613  +1.904096344
 9      +.000221106      +.000042533    +.445922145  -3.468220550  +1.904250896
10      +.000047832      +.000016765    +.445900518  -3.468197008  +1.904215316
xs                                     +.445912469  -3.468208092  +1.904211396
```

$$xiter\_gsv := [0.4459005177,\ -3.468197008,\ 1.904215316]$$

$$iter\_gsv := 10$$

```
> x:=[seq(k,k=0..iter_gsv)]:
  ylog:=[seq(log10(f_gsv[k]/f_gsv[1]),k=1..iter_gsv+1)]:
  if f_gsv[iter_gsv+1]=0 then
   ylog:=[op(1..iter_gsv,ylog),-16]:
  end if:
  pgsv:=[seq([x[k],ylog[k]],k=1..iter_gsv+1)]:

> k:='k':
  pgsv1:=plot(pgsv,k=-0.2..iter_gsv+0.2,color=red,thickness=2):
  pgsv2:=textplot([[5,-0.5,‘log10(||f_gsv[k]||_2/||f_gsv[0]||_2)‘]]):
  plots[display]([pgsv1,pgsv2],axes=boxed,
        title=‘GSV - Logarithmus des relativen Fehlers‘);
```

ESV

```
> dd:="C:/D/Neundorf/Maple3/esv1.res":
  x0:=vector(n,[0$n]);
  etol:=1e-16;
  maxiter:=10;
  f_iv:=vector(maxiter+1,[0$(maxiter+1)]):

  lprint(‘ESV‘):
  erg:=IV(RF,dd,n,A,b,x0,maxiter,etol):
  xiter_esv:=evalf(evalm(erg[1]));
  iter_esv:=erg[2];
  f_esv:=evalm(f_iv):
```

$$x0 := [0,\ 0,\ 0]$$

$$etol := 0.1\,10^{-15}$$

$$maxiter := 10$$

```
ESV
‘Spektralradius  rho(H) = ‘, .9230452958e-1

Iterationsverlauf
 m  ||x(m)-x(m-1)||_2  ||x(m)-xs||_2                  x(m)
 0                     +3.981623582   +0.000000000  +0.000000000  +0.000000000
 1     +4.593882864     +1.122445269   +1.500000000  -3.812500000  +2.078125000
 2     +1.229132052      +.106704123    +.345052083  -3.437337240  +1.888088650
 3      +.116436214      +.009732345    +.455088298  -3.471091800  +1.905696975
 4      +.010635073      +.000902732    +.445060456  -3.467943199  +1.904074171
 5      +.000985893      +.000083161    +.445990924  -3.468232592  +1.904224058
 6      +.000090843      +.000007682    +.445905220  -3.468205833  +1.904210227
 7      +.000008391      +.000000709    +.445913138  -3.468208301  +1.904211503
 8      +.000000774      +.000000065    +.445912407  -3.468208073  +1.904211386
 9      +.000000071      +.000000006    +.445912475  -3.468208094  +1.904211396
10      +.000000007      +.000000001    +.445912469  -3.468208092  +1.904211395
xs                                     +.445912469  -3.468208092  +1.904211396
```

$$xiter\_esv := [0.4459124685,\ -3.468208092,\ 1.904211395]$$
$$iter\_esv := 10$$

Grafik

```
> x:=[seq(k,k=0..iter_esv)]:
  ylog:=[seq(log10(f_esv[k]/f_esv[1]),k=1..iter_esv+1)]:
  if f_esv[iter_esv+1]=0 then
   ylog:=[op(1..iter_esv,ylog),-16]:
  end if:
  pesv:=[seq([x[k],ylog[k]],k=1..iter_esv+1)]:

> k:='k':
  pesv1:=plot(pesv,k=-0.2..iter_esv+0.2,color=blue,thickness=2):
  pesv2:=textplot([[5,-1,‘log10(||f_esv[k]||_2/||f_esv[0]||_2)‘]]):
  plots[display]([pesv1,pesv2],axes=boxed,
        title=‘ESV - Logarithmus des relativen Fehlers‘);
```

Konvergenzrate der 3 IV

```
> iter:=max(iter_rfi,iter_gsv,iter_esv):

> pt:=textplot([[8,-9.6,‘k‘]]):
  pp:=plot([prfi,pgsv,pesv],k=-0.2..iter+0.2,axes=boxed,
            labels=[‘‘,‘‘],thickness=3,
            color=[green,blue,red],linestyle=[DASHDOT,SOLID,DASH],
            title=‘     IV: RF(om),GSV,ESV - relative Fehler
                   log(||x(k)-xs||_2/||x(0)-xs||_2)  ‘,
            legend=[‘RF(om) dashdot‘,‘GSV solid‘,‘ESV dash‘]):

  display(pp,pt);
```

```
> dateiname:=‘iv_01.ps‘:
  ivfile:=cat(‘C:/D/Neundorf/Maple3/‘,dateiname):
  interface(plotdevice=ps,plotoutput=ivfile,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp,pt);
  interface(plotdevice=win);
```

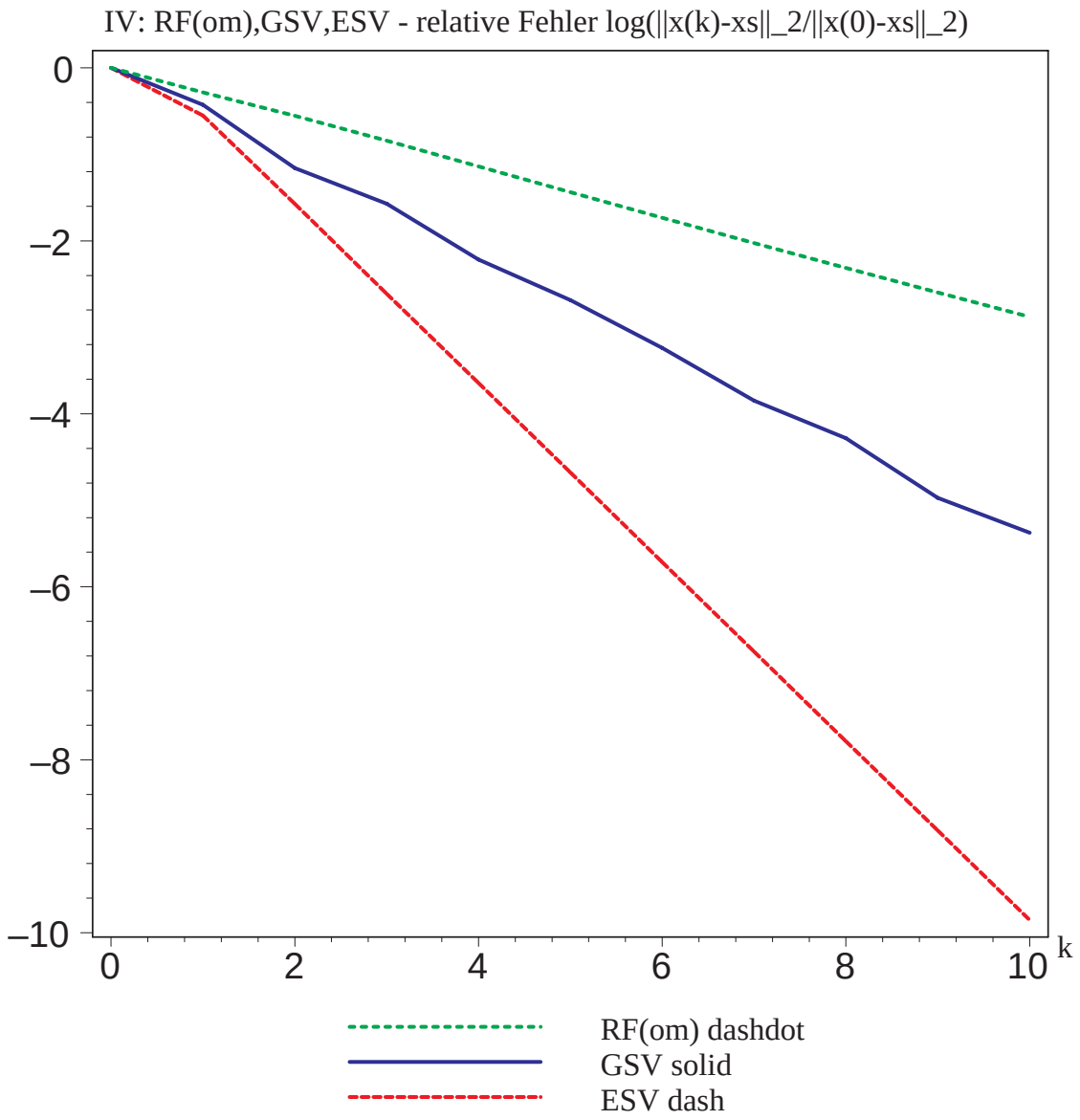


Abb. 3.1 Datei *iv_01.ps*, Konvergenzgeschwindigkeit der 3 IV: RF(ω), GSV, ESV, Darstellung des relativen Fehlers $\log_{10}\left(\frac{\|x^{(k)}-x^*\|_2}{\|x^{(0)}-x^*\|_2}\right),\ k=0,1,...$

Man bemerke wegen $\rho(H_R)^2 \approx \rho(J)$ und $\rho(J)^2 \approx \rho(H_1)$ jeweils die Verdopplung der Konvergenzgeschwindigkeit der IV.

3.3 Fehlerfortpflanzung bei Rekursion und Stabilität

Für die Berechnung des bestimmten Integrals

$$I_n = \frac{1}{e}\int_0^1 e^x x^n dx \ , \quad n = 0, 1, ..., \tag{3.18}$$

mittels Rekursionsformeln nehmen wir zunächst eine Abschätzung von I_n durch Einschachtelung vor.

$$\begin{aligned} ex &\leq e^x \leq e, \quad x \in [0,1] \ , \\ \tfrac{1}{e}\int_0^1 exx^n dx &< I_n < \tfrac{1}{e}\int_0^1 ex^n dx, \\ \tfrac{1}{n+2} &< I_n < \tfrac{1}{n+1}, \end{aligned}$$

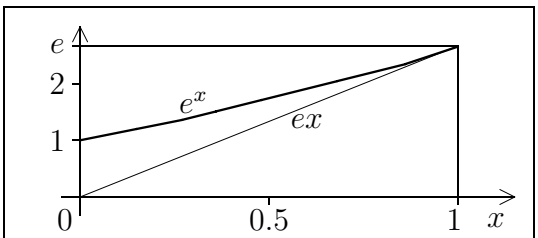


Abb. 3.2 Einschachtelung von e^x

und somit gilt

$$1 > I_0 > \frac{1}{2} > I_1 > \frac{1}{3} > I_2 > ... \ , \quad \lim_{n\to\infty} I_n = 0 \ \text{ und } \ \lim_{n\to\infty} nI_n = 1. \tag{3.19}$$

Durch partielle Integration erhalten wir die typische **lineare Rekursion**

$$\begin{aligned} I_n &= 1 - nI_{n-1}, \ \text{ falls } n > 0, \\ I_0 &= 1 - \frac{1}{e} = 0.63212\,05588\,28557\,67840... \, . \end{aligned} \tag{3.20}$$

Diese kann man auch als Iteration $I_n = 1 - nI_{n-1}$, $n = 1, 2, ...$, I_0 gegeben, interpretieren, zumal theoretisch ja auch ein Grenzwert der Folge $\{I_n\}$ existiert. Durch Entrekursivierung folgt

$$\begin{aligned} I_n &= 1 - n + n(n-1) - n(n-1)(n-2) + ... \\ &\quad +(-1)^n n(n-1)\cdot ... \cdot 4 \cdot 3 - (-1)^n n!/e. \end{aligned} \tag{3.21}$$

Bei Ausführung der Rekursion mit einem Näherungswert für I_0 bemerkt man sehr bald, dass die Einschließung

$$I_n \in \left(\frac{1}{n+2}, \frac{1}{n+1}\right)$$

und damit die Monotonie der Folge verletzt werden.

Die Rekursionsformel ist numerisch instabil. Ein Eingangsfehler, der bei der Näherung der Eulerschen Zahl e gemacht wird, wirkt sich durch die Multiplikation mit $n!$ sehr drastisch aus.
In der GP-Arithmetik in Turbo Pascal mit den Zahlenformaten *double* (15-16 dezimale Mantissenstellen) und *extended* (19-20 Mantissenstellen, 18 Stellen werden angezeigt) sowie mit einem bestmöglichen Startwert I_0, wandert pro Rekursionsschritt die erste ungültige Mantissenstelle um durchschnittlich eine Position nach links.

n	I_n auf 20 Dezimalstellen genau	I_n mit Rekursion und GP-Format	
		double 16 Nachkommastellen	*extended* 18 Nachkommastellen
0	0.63212055882855767840	**0.6321205588285577**	**0.632120558828557678**
1	0.36787944117144232160	**0.3678794411714423**	**0.367879441171442322**
2	0.26424111765711535681	**0.2642411176571153**	**0.264241117657115357**
3	0.20727664702865392957	**0.207276647028654** 0	**0.207276647028653930**
4	0.17089341188538428171	**0.170893411885384** 0	**0.17089341188538428** 1
5	0.14553294057307859146	**0.14553294057308** 01	**0.14553294057307859** 3
6	0.12680235656152845122	**0.1268023565615** 195	**0.1268023565615284** 41
7	0.11238350406930084144	**0.1123835040693** 635	**0.112383504069300** 915
8	0.10093196744559326848	**0.100931967445** 0921	**0.10093196744559** 2678
9	0.09161229298966058367	**0.09161229299** 41707	**0.09161229298966** 5896
10	0.08387707010339416334	**0.0838770700** 582927	**0.0838770701033** 41042
11	0.07735222886266420323	**0.077352229** 3587803	**0.077352228863** 248537
12	0.07177325364802956125	**0.07177324** 76946367	**0.07177325364** 1017554
13	0.06694770257561570369	**0.0669477** 799697233	**0.066947702** 666771802
14	0.06273216394138014834	**0.06273** 10804238732	**0.06273216** 2665194768
15	0.05901754087929777487	**0.0590** 337936419019	**0.0590175** 60022078475
16	0.05571934593123560215	**0.055** 4593017295701	**0.055719** 039646744406
17	0.05277111916899476344	**0.05** 71918705973076	**0.05277** 6326005345098
18	0.05011985495809425803	**-0.0** 294536707515363	**0.050** 026131903788240
19	0.04772275579620909737	1.5596197442791890	**0.04** 9503493828023437
20	0.04554488407581805262	-30.1923948855837807	**0.0** 09930123439531258

Tab. 3.3 Rekursion $I_n = 1 - nI_{n-1}, \quad n = 1, 2, ..., 20, \quad I_0$ maximal genau.

Rechnet man die Rekursionsformel mit Intervallarithmetik und Pascal-XSC mit dem GP-Format *real* (entspricht dem Format *double* in TP), so ist die instabile Fehlerfortpflanzung ähnlich.
Die relative Länge des Startintervalls $[\,\mathrm{I01}\,,\mathrm{I02}\,]$, $\varepsilon = (\mathrm{I02–I01})/\mathrm{I01}$, vergrößert sich ständig für die nachfolgenden Intervalle $[\,\mathrm{I01}\,,\mathrm{I02}\,]^{(n)}$ und die Intervallgrenzen bekommen unbrauchbare Werte.
Nur wenn I01=I02 ist, dann gilt $(\mathrm{I02}^{(n)}–\mathrm{I01}^{(n)})/\mathrm{I01}^{(n)} = \mathcal{O}(10^{-15})$.

Startintervall I_0 : [0.632120 , 0.632121]		
n	[inf(I_n) , sup(I_n)]	
0	[6.32119E-001,	6.32122E-001]
1	[3.67878E-001,	3.67881E-001]
2	[2.64239E-001,	2.64243E-001]
3	[2.0727E-001,	2.0729E-001]
4	[1.708E-001,	1.710E-001]
5	[1.454E-001,	1.457E-001]
6	[1.26E-001,	1.28E-001]
7	[1.1E-001,	1.2E-001]
8	[7.8E-002,	1.2E-001]
9	[–6.9E-002,	3.0E-001]
10	[–2.0E+000,	1.7E+000]
11	[–1.8E+001,	2.3E+001]
12	[–2.7E+002,	2.2E+002]
13	[–2.8E+003,	3.5E+003]
14	[–4.9E+004,	3.9E+004]
15	[–5.8E+005,	7.4E+005]

Startintervall I_0 : [0.63212 , 0.63212]		
n	[inf(I_n) , sup(I_n)]	
0	[6.321199999999999E-001,	6.321200000000001E-001]
1	[3.678799999999999E-001,	3.678800000000001E-001]
2	[2.642399999999998E-001,	2.642400000000001E-001]
3	[2.07279999999999E-001,	2.07280000000001E-001]
4	[1.7087999999999E-001,	1.7088000000001E-001]
5	[1.4559999999999E-001,	1.4560000000002E-001]
6	[1.263999999999E-001,	1.264000000001E-001]
7	[1.15199999999E-001,	1.15200000001E-001]
8	[7.8399999996E-002,	7.8400000001E-002]
9	[2.943999999E-001,	2.944000001E-001]
10	[–1.944000001E+000,	–1.943999999E+000]
11	[2.238399999E+001,	2.238400001E+001]
12	[–2.676080001E+002,	–2.676007999E+002]
13	[3.479903999E+003,	3.479904001E+003]
14	[–4.871765601E+004,	–4.871765599E+004]
15	[7.307658399E+005,	7.307658402E+005]

Tab. 3.4 Rekursion $I_n = 1 - nI_{n-1}$, $n = 1, 2, ..., 15$, I_0 gegeben, mit Pascal-XSC und Intervallarithmetik.

Bemerkung 3.2 Es liegt eine gewisse Verwandtschaft der Rekursion zur Fixpunktiteration auf der Basis von $x = g(x),\ g(x) = 1 - nx$ *abhängig vom Iterationsindex*, $n \geq 1$, vor. Die Instabilität der Rekursion korrespondiert sozusagen mit der Divergenz des allgemeinen Iterationsverfahrens $I_n = g(I_{n-1}),\ g'(I) = -n \leq -1$. Der Fixpunkt wandert mit dem Index von 1/2 nach 0, kann aber wegen wachsenden $|g'| \geq 1$ nicht angenähert werden.

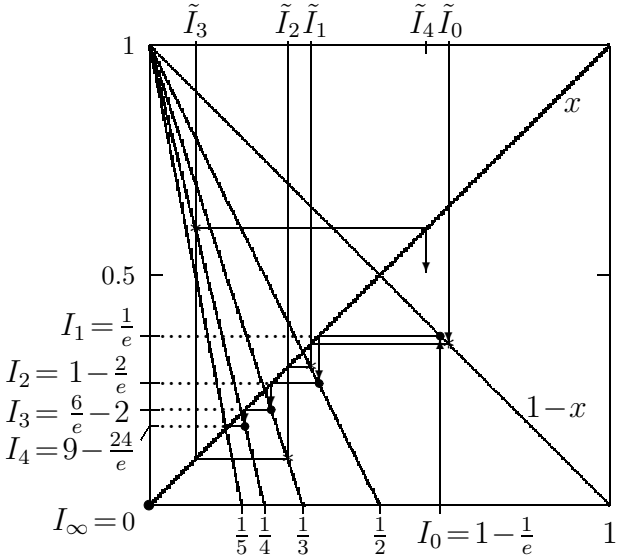


Abb. 3.3
Interpretation der Rekursion $I_n=1-nI_{n-1},\ n=1,2,...,\ I_0=1-\frac{1}{e}$, als "wandernde" Fixpunktiteration, Divergenz ist beim Startwert $\tilde{I}_0 \neq I_0$ sichtbar.

So problematisch und eigentlich nicht anwendbar die Rekursionsformel ist, die Rückwärtsrekursion erweist sich als das positive Gegenteil.
Mit $I_{n-1} = (1 - I_n)/n,\ n = m, m-1, ...1$, könnte man bei einem guten Startwert I_m die vorherliegenden Werte und damit schließlich die Eulersche Zahl ziemlich genau bestimmen.

n	$\tilde{I}_n$ mit Rückwärtsrekursion	
	$\tilde{I}_{10} = 1/12$	$\tilde{I}_{10} = 1/11$
10	**0.083** 3333333333333	**0.0** 909090909090909
9	**0.0916** 666666666667	**0.09** 09090909090909
8	**0.1009** 259259259259	**0.101** 0101010101010
7	**0.11238** 42592592593	**0.1123** 737373737374
6	**0.126802** 2486772487	**0.12680** 37518037518
5	**0.1455329** 585537919	**0.145532** 7080327080
4	**0.17089340** 82892416	**0.1708934** 583934584
3	**0.207276647** 9276896	**0.2072766** 354016354
2	**0.264241117** 3574368	**0.26424112** 15327882
1	**0.367879441** 3212816	**0.36787943** 92336059
0	**0.632120558** 6787184	**0.63212056** 07663941

Tab. 3.5
Rückwärtsrekursion $\tilde{I}_{n-1} = (1 - \tilde{I}_n)/n$, $n = 10, 9, ..., 1$, $\tilde{I}_{10}$ gegeben, Turbo Pascal mit GP-Format *double*.

Was passiert, wenn I_m fehlerbehaftet ist? Wir rechnen also mit $\tilde{I}_m = I_m - \delta_m$.

$$\begin{aligned}
\tilde{I}_{m-1} &= \frac{1-\tilde{I}_m}{m} = \frac{1-(I_m-\delta_m)}{m} = \frac{1-I_m}{m} + \frac{\delta_m}{m} = I_{m-1} - \delta_{m-1}, \\
\delta_{m-1} &= -\frac{\delta_m}{m}, \\
\tilde{I}_{m-2} &= \frac{1-\tilde{I}_{m-1}}{m-1} = \frac{1-(I_{m-1}-\delta_{m-1})}{m-1} = \frac{1-I_{m-1}}{m-1} + \frac{\delta_{m-1}}{m-1} = I_{m-2} - \delta_{m-2}, \\
\delta_{m-2} &= -\frac{\delta_{m-1}}{m-1} = \frac{\delta_m}{(m-1)m} \quad \text{usw.}
\end{aligned}$$

Ein Anfangsfehler δ_m wird bei hinreichend großem m nach wenigen Schritten sehr klein werden. Die Anzahl der Schritte der Vorwärtsrekursion, die alle gültigen Mantissenstellen auslöscht, charakterisiert den Wert m, mit dem die Rückwärtsrekursion bei $\tilde{I}_m \approx 0$ den (genauen) Wert I_0 liefert.

n	$\tilde{I}_n$ mit Rückwärtsrekursion und GP-Format	
	double 16 Nachkommastellen	*extended* 18 Nachkommastellen
20		**0.** 000000000000000000
19		**0.05** 0000000000000000
18		**0.050** 000000000000000
17	**0.** 0000000000000000	**0.05277** 7777777777778
16	**0.05** 88235294117647	**0.055718** 954248366013
15	**0.058** 8235294117647	**0.0590175** 65359477124
14	**0.0627** 450980392157	**0.06273216** 2309368192
13	**0.06694** 67787114846	**0.066947702** 692187986
12	**0.071773** 3247145012	**0.0717732536** 39062463
11	**0.07735222** 29404582	**0.07735222886** 3411462
10	**0.083877070** 6417765	**0.083877070103** 326231
9	**0.0916122929** 358223	**0.0916122929896** 67377
8	**0.1009319674** 515753	**0.10093196744559** 2514
7	**0.11238350406** 85531	**0.112383504069300** 936
6	**0.126802356561** 6353	**0.1268023565615284** 38
5	**0.1455329405730** 608	**0.14553294057307859** 4
4	**0.17089341188538** 78	**0.17089341188538428** 1
3	**0.20727664702865** 30	**0.207276647028653930**
2	**0.264241117657115** 7	**0.264241117657115357**
1	**0.367879441171442** 2	**0.367879441171442322**
0	**0.632120558828557** 9	**0.632120558828557678**

Tab. 3.6 Rückwärtsrekursion $\tilde{I}_{n-1} = (1-\tilde{I}_n)/n,\ n = m(-1)1,\ \tilde{I}_m = 0,$ $m = 17, 20,$ Turbo Pascal.

Bei Rückwärtsrekursion wird mit wachsendem Index m zuerst $\tilde{I}_0$ immer genauer bis die Genauigkeit eine immer größere Anzahl von anfänglichen Werten erfasst.

m	$\tilde{I}_0$	$\delta_0 = I_0 - \tilde{I}_0$
2	0.5	0.13
4	0.625	0.0071
6	0.63194	0.00017
8	0.63211 805	0.00000 25
10	0.63212 05357	0.00000 0023
12	0.63212 05586 78718	0.00000 00001 5
14	0.63212 05586 27838	0.00000 00000 0072
16	0.63212 05586 28555	0.00000 00000 00003
17	0.63212 05586 28558	0

Tab. 3.7 Verbesserung der Genauigkeit bei Rückwärtsrekursion $\tilde{I}_{n-1} = (1 - \tilde{I}_n)/n, \quad n = m, m-1, ..., 1, \quad \tilde{I}_m = 0,$ mit wachsendem Startindex m, Turbo Pascal mit GP-Format *double*.

m	k	$\delta_k = I_k - \tilde{I}_k$
20	4	1E-18
21	6	–1E-18
22	7	–1E-18
23	9	1E-18
24	11	–3E-18
25	12	–1E-18
26	14	7E-18
27	14	–1E-18
28	16	2E-18
29	17	1E-18
30	17	–1E-18
31	20	1E-18
32	20	1E-18
33	21	

Tab. 3.8
Vergrößerung der Mengen der genauen Werte $\tilde{I}_0, \tilde{I}_1, ..., \tilde{I}_{k-1}$ bei Rückwärtsrekursion mit wachsendem m, $\tilde{I}_m = 0$, Turbo Pascal mit GP-Format *extended*.

Bezüglich des stark gedämpften Fehlers betrachten wir noch einige Abschätzungen. Für die Werte $\{I_m, \tilde{I}_m\}$ gilt

$$\frac{1}{m+2} < I_m < \frac{1}{m+1} < I_{m-1} < \frac{1}{m} < I_{m-2} < \frac{1}{m-1} < ... \tag{3.22}$$

und mit $\tilde{I}_m = 0$

$$0 = \tilde{I}_m < \frac{1}{m+1} < \tilde{I}_{m-1} = \frac{1}{m} = \tilde{I}_{m-2} < \frac{1}{m-1} < ...$$

und weiter

$$\begin{aligned} |I_m - \tilde{I}_m| &< \frac{1}{m+1} = \mathcal{O}(1/m), \\ |I_{m-1} - \tilde{I}_{m-1}| &= \frac{1}{m} - \frac{1}{m+1} = \frac{1}{m(m+1)} = \mathcal{O}(1/m^2), \\ |I_{m-2} - \tilde{I}_{m-2}| &= \frac{1}{m-1} - \frac{1}{m} = \frac{1}{(m-1)m}. \end{aligned}$$

Die wachsende Größenordnung der nächsten Verbesserung ist in der Abschätzung nicht zu erkennen. Einen Hinweis darauf erhält man jedoch schon, wenn man z. B. mit der Annahme $I_m = 1/(m + \frac{3}{2})$ rechnet. Daraus ergeben sich

$$I_{m-1} = \frac{2m+1}{m(2m+3)}, \quad I_{m-2} = \frac{2m^2+m-1}{(m-1)m(2m+3)},$$

und somit

$$|I_{m-2} - \tilde{I}_{m-2}| = \frac{2m^2+m-1}{(m-1)m(2m+3)} - \frac{1}{m} = \frac{2}{(m-1)m(2m+3)} = \mathcal{O}(1/m^3).$$

Wir fassen die wichtigsten Schlussfolgerungen der Rückwärtsrekursion zusammen:

- Je größer m ist oder je genauer $\tilde{I}_m$, desto kleiner $|I_0 - \tilde{I}_0|$ bzw. desto mehr Anfangselemente $\tilde{I}_0, \tilde{I}_1, ..., \tilde{I}_{k-1}$ sind genau.
- Je größer m ist, desto näher "rückt" der erste genaue Wert $\tilde{I}_{k-1}$.
- Bei $m = 10^5$ und $\tilde{I}_m = 0$ gilt für den relativen Fehler $\varepsilon_m = |(I_m - \tilde{I}_m)/I_m|$

$$\begin{aligned} \varepsilon_m &= 1, \\ \varepsilon_{m-1} &= \mathcal{O}(1/m) = \mathcal{O}(10^{-5}), \\ \varepsilon_{m-2} &= \mathcal{O}(1/m^2) = \mathcal{O}(10^{-10}), \\ \varepsilon_{m-3} &= \mathcal{O}(10^{-15}), \end{aligned}$$

so dass man bei 15–16 Mantissenstellen nach 3 Rekursionsschritten den "genauen Bereich" erreicht. Mittels dieser Rückwärtsrekursion kann man mit geringem Aufwand für jedes n das Integral I_n mit gegebener Maschinengenauigkeit berechnen.

- In der Intervallarithmetik ist die Stabilität noch ausgeprägter.
  Vergleicht man die Rückwärtsrekursion von Tab. 3.6 ($\tilde{I}_{20} = 0,$ *extended*) mit den (genauen) Werten $\tilde{I}_0, \tilde{I}_1, \tilde{I}_2, \tilde{I}_3, (\tilde{I}_4)$ und die der nachfolgenden Tabelle 3.9 mit dem Startintervall $\tilde{I}_{20} = [0, 0.09] \approx [0, 2I_{20}]$, so bemerkt man ein weiteres Anwachsen der Zahl der genauen Werte (hier $\tilde{I}_0, ..., \tilde{I}_6$).

n	Startintervall $\tilde{I}_{20}$	
	[0,	0.09]
20	[0.0E-000,	9.1E-002]
19	[4.5E-002,	5.1E-002]
18	[4.99E-002,	5.03E-002]
17	[5.276E-002,	5.278E-002]
16	[5.5718E-002,	5.5720E-002]
15	[5.901751E-002,	5.901757E-002]
14	[6.2732162E-002,	6.2732166E-002]
13	[6.69477024E-002,	6.69477027E-002]
12	[7.177325363E-002,	7.177325366E-002]
11	[7.7352228861E-002,	7.7352228864E-002]
10	[8.38770701033E-002,	8.38770701035E-002]
9	[9.161229298965E-002,	9.161229298967E-002]
8	[1.0093196744559E-001,	1.0093196744560E-001]
7	[1.12383504069300E-001,	1.12383504069301E-001]
6	[1.268023565615284E-001,	1.268023565615285E-001]
5	[1.455329405730785E-001,	1.455329405730787E-001]
4	[1.708934118853842E-001,	1.708934118853844E-001]
3	[2.072766470286539E-001,	2.072766470286540E-001]
2	[2.642411176571153E-001,	2.642411176571154E-001]
1	[3.678794411714422E-001,	3.678794411714424E-001]
0	[6.321205588285576E-001,	6.321205588285578E-001]

Tab. 3.9 Rückwärtsrekursion
$\tilde{I}_{n-1} = (1 - \tilde{I}_n)/n, \quad n = 20, 19, ..., 1, \quad \tilde{I}_{20} = [0, 0.9]$,
mit Pascal-XSC und Intervallarithmetik.

Rechnungen in Maple

Fehlerfortpflanzung bei Rekursion und Stabilitaet
Rekursive Berechnung von bestimmten Integralen

```
> n:='n':
  x:='x':
  f:=x->exp(x)*x^n;

  Int(f(x),x):
  int(f(x),x):                          # Verwendung der Gamma-Funktion
  %%=%;
  `In`=1/exp(1)*int(f(x),x=0..1);   # Int->0 bei n->unendlich
  limit(int(f(x),x=0..1),n=infinity)=`0`;
```

$$f := x \to e^x\, x^n$$

$$\int e^x x^n\, dx =$$

$$-(-1)^{(-n)}(x^n\,(-1)^n\,n\,\Gamma(n)\,(-x)^{(-n)} - x^n\,(-1)^n e^x - x^n\,(-1)^n\,n\,(-x)^{(-n)}\,\Gamma(n,-x))$$

$$In = \frac{1}{e}\int_0^1 e^x x^n\, dx$$

$$\lim_{n\to\infty}\int_0^1 e^x x^n\, dx = 0$$

Grafik zum Verhalten des bestimmten Integrals (Flaeche)

```
> ff:=(x,n)->exp(x-1)*x^n:
  i:='i':
  m:=10:
  p1:=array(0..m,[]):
  for i from 0 to m do
   p1[i]:=plot(ff(x,i),x=0..1,color=black,thickness=1);
  end do:
  p2:=plot(ff(x,3),x=0..1,filled=true,color=yellow):
  p21:=plot(ff(x,3),x=0..1,color=black,thickness=3):
  p3:=plot([[1,0],[1,1]], color=black,thickness=3):
  p4:=textplot([[0.5,0.65,`n=0`],[0.5,0.36,`n=1`],[0.7,0.31,`n=3`]]):
  pp:=plots[display]([p2,p21,p3,p4,seq(p1[i],i=0..m)],
      title=`   Flaeche als bestimmtes Integral
             von f(x)=exp(x-1)x^n ueber [0,1]`):
  display(pp);

> pfad:=`C:/D/Neundorf/Maple3/`:
  dateiname:=`int_01`:
  file:=cat(pfad,dateiname,`.ps`):
  interface(plotdevice=ps,plotoutput=file,
            plotoptions=`portrait,noborder`);
            # Standard ist Dim.=(414,560)pt mit Transl.=(72,72)pt
            # Bildgroesse ca. 145.5*196.8mm
  plots[display](pp);
  interface(plotdevice=win);
```

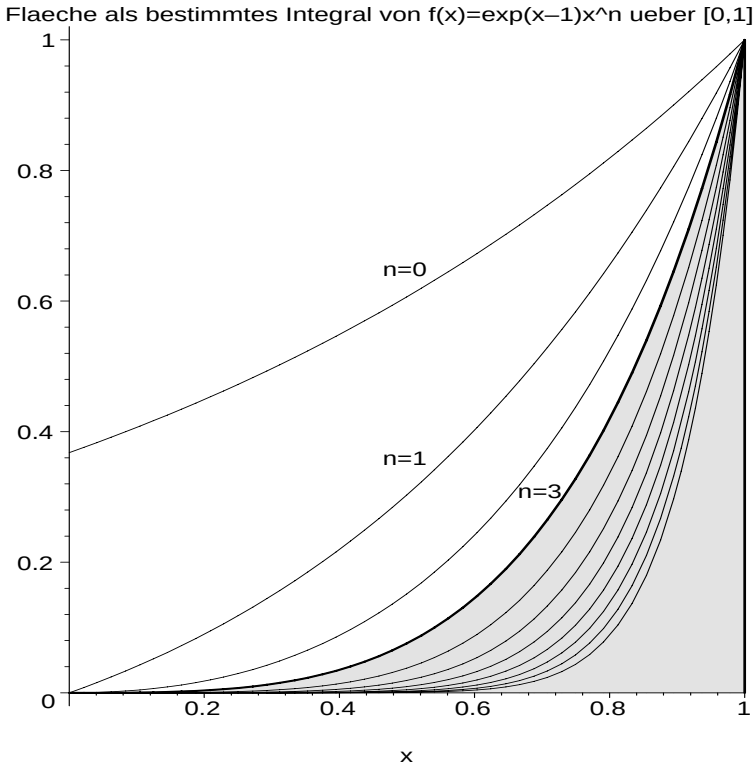


Abb. 3.4
Datei *int_01.ps*
Fläche als
$\int_0^1 e^{x-1}x^n\,dx$,
$n = 0(1)10$

Rekursive Prozeduren fuer I_n

```
> In := proc(n) option remember;
       # n ganzzahlig , aber auch reell
   if round(n)=0 then 1-1/exp(1) else 1-n*In(n-1) end if;
  end:

> In(0), In(1), In(2), In(3);
```

$$1-\frac{1}{e},\ \frac{1}{e},\ 1-\frac{2}{e},\ -2+\frac{6}{e}$$

```
> In1 := proc(n)
   if round(n)>1 then In1(n):=1-n*In1(n-1)
                 else In1(n):=1-n*In1(0)
   end if;
  end:
  In1(0):=1-1/exp(1);
```

$$In1(0) := 1-\frac{1}{e}$$

```
> In1(0), In1(1), In(2), In(3);
```

$$1-\frac{1}{e},\ \frac{1}{e},\ 1-\frac{2}{e},\ -2+\frac{6}{e}$$

Test von Arithmetik und Stellenausloeschung.
Man beachte die abweichenden Ergebnisse der Prozeduren `In` und `In1` mit reellem bzw. ganzzahligem Argument fuer grosse n.

```
> Digits:=16;   #  6,10,16,20,40
  n:=20;        # 10,20, ...
  xn:=n*1.0;
  In(n)=evalf(In(n));
  In(xn)=evalf(In(xn));
  Digits:=20;
  In(n)=evalf(In(n));
  In(xn)=evalf(In(xn));
```

$$Digits := 16$$

$$n := 20$$

$$xn := 20.0$$

$$89501463119290212 1 - \frac{2432902008176640000}{e} = -200.$$

$$0.8950146311929022\,10^{18} - \frac{0.2432902008176640\,10^{19}}{e} = -100.$$

$$Digits := 20$$

$$89501463119290212 1 - \frac{2432902008176640000}{e} = 0.06$$

$$0.8950146311929022\,10^{18} - \frac{0.2432902008176640\,10^{19}}{e} = 79.06$$

```
> Digits:=16;   #  6,10,16,20,40
  n:=20;        # 10,20, ...
  xn:=n*1.0;
  In1(n)=evalf(In1(n));
  In1(xn)=evalf(In1(xn));
  Digits:=20;
  In1(n)=evalf(In1(n));
  In1(xn)=evalf(In1(xn));
```

$$Digits := 16$$

$$n := 20$$

$$xn := 20.0$$

$$89501463119290212 1 - \frac{2432902008176640000}{e} = -200.$$

$$0.8950146311929022\,10^{18} - \frac{0.2432902008176640\,10^{19}}{e} = -100.$$

$$Digits := 20$$

$$89501463119290212 1 - \frac{2432902008176640000}{e} = 0.06$$

$$0.8950146311929022\,10^{18} - \frac{0.2432902008176640\,10^{19}}{e} = 79.06$$

Achtung: ungueltige Stellen durch 0 am Ende zu erkennen.
Man beachte auch hier die abweichenden Ergebnisse der Prozeduren mit reellem bzw. ganzzahligem Argument im Ueberlaufbereich (fuer grosses i bzw. ix)

```
> Digits:=20;    # Teste 22, 24, 40, 60,  ...
  n:=25;
  In1(0):=1-1/exp(1.0);
  `In(0)`=In(0);
  printf(` `):
  printf(` i                In1(ix)          In(ix)           In(i)\n`):
  printf(`%2d   %24.16f  %27.16f
          %24.16f\n`,0,In1(0.0),evalf(In(0.0)),evalf(In(0))):
  i:='i':
  for i from 1 to n do
   ix:=1.0*i:
   printf(`%2d   %24.16f  %27.16f
           %24.16f\n`,i,evalf(In1(ix)),evalf(In(ix)),evalf(In(i)))
  end do:
```

$$Digits := 20$$
$$n := 25$$
$$In1(0) := 0.63212055882855767841$$
$$In(0) = 1 - \frac{1}{e}$$

```
 i              In1(ix)                    In(ix)                 In(i)
 0         1.0000000000000000          .6321205588285577       .6321205588285577
 1          .3678794411714423          .3678794411714423       .3678794411714423
 2          .2642411176571154          .2642411176571154       .2642411176571154
 3          .2072766470286539          .2072766470286539       .2072766470286539
 4          .1708934118853843          .1708934118853843       .1708934118853843
 5          .1455329405730786          .1455329405730786       .1455329405730786
 6          .1268023565615285          .1268023565615285       .1268023565615285
 7          .1123835040693008          .1123835040693008       .1123835040693008
 8          .1009319674455930          .1009319674455930       .1009319674455930
 9          .0916122929896600          .0916122929896600       .0916122929896600
10          .0838770701034000          .0838770701034000       .0838770701034000
11          .0773522288620000          .0773522288620000       .0773522288620000
12          .0717732536500000          .0717732536500000       .0717732536500000
13          .0669477025000000          .0669477025000000       .0669477025000000
14          .0627321640000000          .0627321640000000       .0627321640000000
15          .0590175300000000          .0590175300000000       .0590175300000000
16          .0557195000000000          .0557195000000000       .0557195000000000
17          .0527700000000000          .0527700000000000       .0527700000000000
18          .0502000000000000          .0502000000000000       .0502000000000000
19        -3.9530000000000000        -3.9530000000000000       .0470000000000000
20        79.0600000000000000        79.0600000000000000       .0600000000000000
21     -1659.0000000000000000     -1659.0000000000000000      0.0000000000000000
22     36510.0000000000000000     36510.0000000000000000     10.0000000000000000
23   -839600.0000000000000000   -839600.0000000000000000   -200.0000000000000000
24  20150000.0000000000000000  20150000.0000000000000000      0.0000000000000000
25 -503700000.0000000000000000 -503700000.0000000000000000 -100000.0000000000000000
```

3.4 Subtraktionskatastrophe bei Bestimmung von π nach Archimedes

Durch iterative Bestimmung der Umfänge von ein- und umbeschriebenen regelmäßigen Vielecken im Kreis kann man die Kreiszahl π einschachteln.
Die Seitenlänge s_{2n} des einbeschriebenen $2n$-Ecks im Kreis mit dem Radius $r = 1$ berechnet sich mittels Rekursion aus der Seitenlänge s_n des n-Ecks.

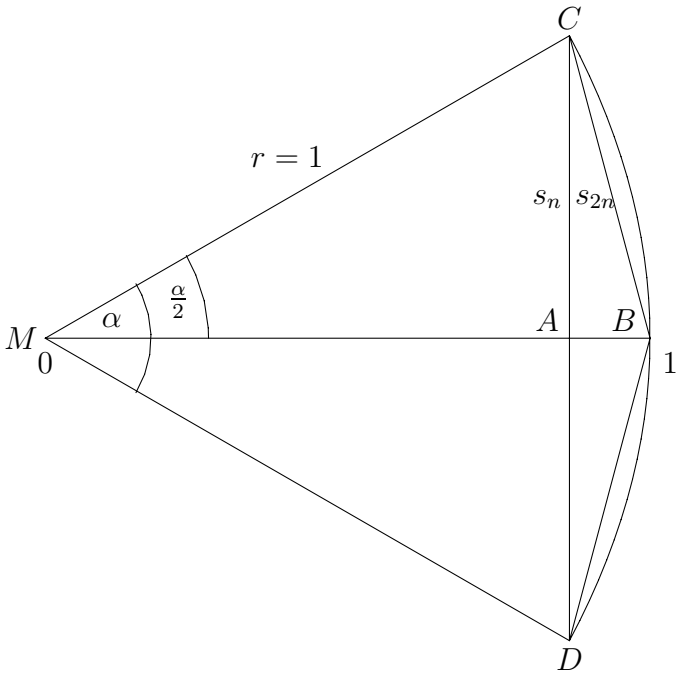


Abb. 3.5
Datei *pi1.pic*
Ausschnitt aus Situation zwischen 6-Eck und 12-Eck im Kreis, $n = 6$, $\alpha = \frac{\pi}{3}$

Es gelten in den rechtwinkeligen Dreiecken MAC und CAB die folgenden Beziehungen.

$$
\begin{aligned}
U_n &= n s_n, \quad \text{Umfang des } n\text{-Ecks}, \quad s_n = \overline{CD}, \ \alpha \text{ Dreiecksinnenwinkel},\\
U_{2n} &= 2n s_{2n}, \quad s_{2n} = \overline{CB},\\
1 &= (s_n/2)^2 + x^2, \quad x = \overline{MA},\\
s_{2n}^2 &= (s_n/2)^2 + (1-x)^2,\\
x &= 1 - s_{2n}^2/2,\\
0 &= s_{2n}^4 - 4s_{2n}^2 + s_n^2, \quad z = s_{2n}^2,\\
0 &= z^2 - 4z + s_n^2,\\
z &= s_{2n}^2 \ = \ 2 \pm \sqrt{4 - s_n^2}, \quad \text{Vorz. } + \text{ ist auszuschließen},\\
s_{2n} &= \sqrt{2 - \sqrt{4 - s_n^2}}.
\end{aligned}
$$

Somit ergibt sich die Rekursionsformel zwischen den Seitenlängen in zwei Varianten

$$s_{2n} \;=\; \sqrt{2-\sqrt{4-s_n^2}} \;=\; \frac{s_n}{\sqrt{2+\sqrt{4-s_n^2}}},\; n=6,12,...,\; s_6=1, \tag{3.23}$$

aus denen die Kreiszahl $\pi \approx \frac{1}{2}\, U_{2n} = n s_{2n}$ folgt.
Die algebraisch gleichwertigen Formeln für s_{2n} verhalten sich numerisch sehr verschieden. Wegen $\lim\limits_{n\to\infty} s_n = 0$ führt die erstere zu einer Subtraktion von annähernd gleichen Zahlen im Computer (mit endlicher Mantissenlänge des GP-Formats) und damit zu unerwünschten und katastrophalen Stellenauslöschung.
Nicht nur in diesem Fall lässt sich die Subtraktionskatastrophe durch eine geeignete Termumformung umgehen.

	Variante 1		Variante 2	
j	$s_{2n}=\sqrt{2-\sqrt{4-s_n^2}}$ $n=6\cdot 2^j$	$\pi^{(1)}$	$s_{2n}=\frac{s_n}{\sqrt{2+\sqrt{4-s_n^2}}}$	$\pi^{(2)}$
0	1.0000000000E-00	3.0000000000	1.0000000000E-00	3.0000000000
1	5.1763809020E-01	3.1058285412	5.1763809020E-01	3.1058285412
2	2.6105238444E-01	3.1326286132	2.6105238444E-01	3.1326286133
3	1.3080625846E-01	3.1393502029	1.3080625846E-01	3.1393502030
4	6.5438165642E-02	3.1410319508	6.5438165643E-02	3.1410319509
5	3.2723463242E-02	3.1414524712	3.2723463253E-02	3.1414524723
6	1.6362279155E-02	3.1415575977	1.6362279208E-02	3.1415576079
7	8.1812079465E-03	3.1415838514	8.1812080524E-03	3.1415838921
8	4.0906125332E-03	3.1415904255	4.0906125823E-03	3.1415904632
9	2.0453070448E-03	3.1415916208	2.0453073607E-03	3.1415921060
10	1.0226528554E-03	3.1415895717	1.0226538140E-03	3.1415925167
11	5.1132598302E-04	3.1415868397	5.1132692372E-04	3.1415926194
12	2.5566299151E-04	3.1415868397	2.5566346395E-04	3.1415926450
13	1.2782793831E-04	3.1414994119	1.2783173224E-04	3.1415926514
14	6.3903295775E-05	3.1409747940	6.3915866151E-05	3.1415926530
15	3.1944530915E-05	3.1402751671	3.1957933079E-05	3.1415926534
16	1.5958023577E-05	3.1374750995	1.5978966540E-05	3.1415926535
17	7.9790117887E-06	3.1374750995	7.9894832701E-06	3.1415926535
18	3.8146972656E-06	3.0000000000	3.9947416351E-06	3.1415926536
19	1.9073486328E-06	3.0000000000	1.9973708175E-06	3.1415926536
20	0.0000000000E-00	0.0000000000	9.9868540877E-07	3.1415926536
21	0.0000000000E-00	0.0000000000	4.9934270438E-07	3.1415926536

Tab. 3.10 π nach Archimedes mit einbeschriebenen Vielecken,
TP mit GP-Format *real*, 6 Byte, 11-12 gültige Dezimalen der Mantisse.

Rechnungen in Maple

Kreiszahl π nach Archimedes mit einbeschriebenen regelmaessigen Vielecken

```
> p1:=implicitplot(x^2+y^2=1,x=-1..1,y=-1..1,scaling=constrained,
      thickness=2,
      title=‘   Kreiszahl Pi nach Archimedes mit
                einbeschriebenen Vielecken‘):

  l12:=[seq([cos(Pi*t/6),sin(Pi*t/6)],t=0..12)]:
  p2:=plot(l12,color=blue):
  l6:=[seq([cos(Pi*t/3),sin(Pi*t/3)],t=0..6)]:
  p3:=plot(l6,color=green):
  p4:=plot([[0,0.005],[1,0.005],[cos(Pi/6),sin(Pi/6)+0.01],[0,0],
           [cos(Pi/6),-sin(Pi/6)],[cos(Pi/6),sin(Pi/6)],[1,0.005],
           [cos(Pi/6),-sin(Pi/6)+0.01]],color=black,thickness=4):
  p5:=textplot([[0.5,0.38,‘r=1‘]]):

  plots[display](p1,p2,p3,p4,p5);

> dateiname:=‘pi_01.ps‘:
  pifile:=cat(‘C:/D/Neundorf/Maple3/‘,dateiname):

  interface(plotdevice=ps,plotoutput=pifile,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](p1,p2,p3,p4,p5);
  interface(plotdevice=win);
```

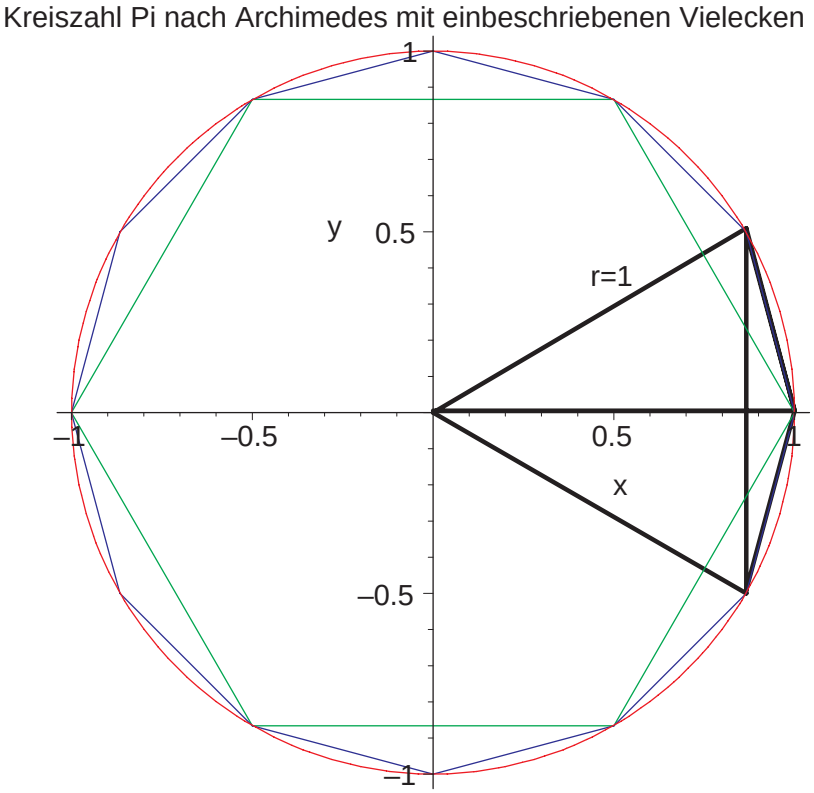


Abb. 3.6
Datei *pi_01.ps*
π nach Archimedes mit einbeschriebenen 6- und 12-Eck

Anwendung der 2 Rekursionen fuer die Berechnung der Vieleckseite und daraus π

```
> Digits:=12:    # Teste 22, 24, ...
  n:=21:
  s1:=1.0: Pi1:=3*s1: s2:=1.0: Pi2:=3*s2:
  lprint(‘2 Rekursionsformeln fuer Seitenlaenge: Digits=12‘):
  lprint(‘1. mit Stellenausloeschung, 2. ohne‘):
  fprintf(default,‘ ‘):
  fprintf(default,
  ‘  i             s1            Pi1            s2            Pi2\n‘):
  fprintf(default,‘ %2d        %12.10f %13.10f  %13.10f  %13.10f\n‘,
                 0,s1,Pi1,s2,Pi2):
  i:=’i’:
  for i from 1 to n do
   s1:=sqrt(2-sqrt(4-s1^2)):    Pi1:=3*2^i*s1:
   s2:=s2/sqrt(2+sqrt(4-s2^2)): Pi2:=3*2^i*s2:
   fprintf(default,‘ %2d        %12.10f %13.10f  %13.10f   %13.10f\n‘,
                  i,s1,Pi1,s2,Pi2):
  end do:
```

```
2 Rekursionsformeln fuer Seitenlaenge: Digits=12
1. mit Stellenausloeschung, 2. ohne
```

i	s1	Pi1	s2	Pi2
0	1.0000000000	3.0000000000	1.0000000000	3.0000000000
1	.5176380902	3.1058285412	.5176380902	3.1058285412
2	.2610523844	3.1326286133	.2610523844	3.1326286133
3	.1308062585	3.1393502028	.1308062585	3.1393502030
4	.0654381656	3.1410319499	.0654381656	3.1410319509
5	.0327234633	3.1414524763	.0327234633	3.1414524723
6	.0163622792	3.1415576028	.0163622792	3.1415576079
7	.0081812077	3.1415837702	.0081812081	3.1415838921
8	.0040906124	3.1415903413	.0040906126	3.1415904632
9	.0020453068	3.1415912801	.0020453074	3.1415921060
10	.0010226534	3.1415912801	.0010226538	3.1415925167
11	.0005113316	3.1416213194	.0005113269	3.1415926193
12	.0002556756	3.1417414740	.0002556635	3.1415926450
13	.0001278280	3.1415011602	.0001278317	3.1415926514
14	.0000639531	3.1434231556	.0000639159	3.1415926530
15	.0000319374	3.1395779882	.0000319579	3.1415926535
16	.0000161245	3.1702087428	.0000159790	3.1415926536
17	.0000083666	3.2898810899	.0000079895	3.1415926536
18	.0000044721	3.5170308234	.0000039947	3.1415926536
19	.0000031623	4.9738326897	.0000019974	3.1415926536
20	0.0000000000	0.0000000000	.0000009987	3.1415926536
21	0.0000000000	0.0000000000	.0000004993	3.1415926536

3.5 Termumformung und mathematische Äquivalenz von Formeln

Empfiehlt man dem Betrachter für die Berechnung eines Wertes die mathematisch äquivalenten Formeln

$$\begin{aligned}
&(1) \quad \frac{1}{(3+\sqrt{10})^4} \;=\; 6.93481\,60951\,23042\,52272...\mathrm{E}-4,\\
&(2) \quad (3-\sqrt{10})^4,\\
&(3) \quad (19-6\sqrt{10})^2,\\
&(4) \quad 721-228\sqrt{10},\\
&(5) \quad \frac{1}{(19+6\sqrt{10})^2},\\
&(6) \quad \frac{1}{721+228\sqrt{10}},
\end{aligned} \tag{3.24}$$

so wird er auf den ersten Blick vom ästhetischen Standpunkt aus sich für die Verwendung der Formel (3) entscheiden. Dass er damit fast den numerisch schlechtesten Ausdruck gewählt hat, wird verursacht durch das schon erwähnte Subtraktionsdilemma.
Im folgenden Turbo-Pascal-Programm mit wahlweiser "künstlicher" Festlegung der Mantissenlänge auf 2...5 Byte kann man die "Güte" der Formeln schnell herausfinden.

```
program Aequivalenz;
{von Quadratwurzel-Bruchtermen bei kuenstlich gekuerzter Mantisse}

uses crt;

type float = real;

var  arr   : array[0..5] of byte;
     x     : float absolute arr;
     n,m   : integer;

{kuenstliches Abschneiden der Mantisse auf n-1 Bytes}
procedure clear;
var i:integer;
begin
  for i:=1 to 6-n do arr[i]:=0
end;

begin
  clrscr;
  writeln('Aequivalenz von Termen');
  writeln;
```

```
  writeln(
   'Byteanzahl der GKZ/Laenge der Mantisse/gueltige Dez.stellen');
  writeln('  n = 3 / 2 / 4                ');
  writeln('      4 / 3 / 7   Format single');
  writeln('      5 / 4 / 9                ');
  writeln('      6 / 5 / 11  Format real  ');
  writeln;
  write('  n = ');
  readln(n);
  writeln;
  case n of  { m --> Format m+6,  6 = Vorzeichen & Punkt & E+00 }
    3 : m:=4;
    4 : m:=7;
    5 : m:=9;
    6 : m:=11
  end;
  {Formel (1), Formeln (2),...,(6) analog}
  x:=10;      clear;
  x:=sqrt(x); clear;
  x:=3+x;     clear;
  x:=sqr(x);  clear;
  x:=sqr(x);  clear;
  x:=1/x;     clear;
  write(' (1)  ',x:m+6);
  x:=1/sqr(sqr(3+sqrt(10)));
  writeln('    real  : ',x);
  writeln;
  writeln(' Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)');
  writeln;
  writeln('     exakt :  6.93481609512304252272...E-04');
  readln
end.
```

Hier sind die Ergebnisse zur numerischen Äquivalenz von Termen mit obigen Turbo-Pascal-Programm im Vergleich der GP-Formate *single* und *real*.

```
Aequivalenz von Termen
Byteanzahl der GKZ/Laenge der Mantisse/gueltige Dez.stellen
  n = 3 / 2 / 4
      4 / 3 / 7   Format single
      5 / 4 / 9
      6 / 5 / 11  Format real
  n = 4
 (1)   6.934818E-04    real  :  6.9348160951E-04
 (2)   6.934781E-04    real  :  6.9348160952E-04
 (3)   6.936202E-04    real  :  6.9348160973E-04
 (4)   7.934570E-04    real  :  6.9348141551E-04
 (5)   6.934817E-04    real  :  6.9348160951E-04
 (6)   6.934816E-04    real  :  6.9348160951E-04
 Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)
      exakt :  6.93481609512304252272...E-04
```

Mit wachsender Mantisse können auch mit den problematischen Formeln (2), (3), (4) bessere Ergebnisse erzielt werden, wie bei Turbo Pascal mit GP-Format *extended*.

```
Aequivalenz von Termen in TP7.0/BP7.0

Format extended = 10Byte => 19-20stellige Mantisse

 (1) :  6.93481609512304252E-0004
 (2) :  6.93481609512304252E-0004
 (3) :  6.93481609512304262E-0004
 (4) :  6.93481609512325292E-0004
 (5) :  6.93481609512304252E-0004
 (6) :  6.93481609512304252E-0004

 Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)

      exakt :  6.93481609512304252272...E-04
```

Natürlich kann man in Turbo Pascal analoge Rechnungen mit den GP-Formaten *single*, *real* oder *double* durchführen. Dabei ist jedoch folgendes zu beachten. Arithmetische Ausdrücke werden bei ihrer Berechnung mit der Stellenzahl der *extended*-Mantisse behandelt (Rechnung jedoch nicht unbedingt in dieser Gemauigkeit) und erst durch Zuordnung eines Wertes zu einer Variablen kommt es zur Stellenabschneidung.
Bei komplizierten Formeln müsste man also auch "zwischendurch" immer wieder die Mantisse kürzen (siehe TP-Programm).
Es ist also ein Unterschied zwischen den ausgegebenen Größen bei folgender Anweisungsfolge

```
...
var x:single;
...
x:=721-228*sqrt(10);
writeln(x);                    { --> 6.93481590.........E-004 }

writeln(721-228*sqrt(10):26);  { --> 6.93481609512325292E-004 }
...
```

Zum Vergleich sei noch eine Rechnung mit Pascal-XSC mit dem GP-Format *real* angegeben.

```
Aequivalenz von Termen in Pascal-XSC
Format real = 8Byte => 15-16stellige Mantisse

 (1) :  6.934816095123040E-004
 (2) :  6.934816095123075E-004
 (3) :  6.934816095122907E-004
 (4) :  6.934816094599228E-004
 (5) :  6.934816095123043E-004
 (6) :  6.934816095123043E-004

 Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)

      exakt :  6.93481609512304252272...E-04
```

Rechnungen in Maple

Termumformung und mathematische Aequivalenz von Formeln

```
> Digits:=30;
  t1 := 1/(3+sqrt(10))^4;
  evalf(t1);
```

$$Digits := 30$$

$$t1 := \frac{1}{(3+\sqrt{10})^4}$$

$$0.000693481609512304252271869340176$$

```
> Digits:=7;

  t1 := 1/(3+sqrt(10.0))^4;
  t2 := (3-sqrt(10.0))^4;
  t3 := (19-6*sqrt(10.0))^2;
  t4 := 721-228*sqrt(10.0);
  t5 := 1/(19+6*sqrt(10.0))^2;
  t6 := 1/(721+228*sqrt(10.0));
```

$$Digits := 7$$

$$t1 := 0.0006934815$$
$$t2 := 0.0006934874$$
$$t3 := 0.0006932689$$
$$t4 := 0.0006$$
$$t5 := 0.0006934815$$
$$t6 := 0.0006934818$$

```
> i:='i':
  n:=16:

  printf(‘Tabelle mit Genauigkeiten  Digits=1..16,
          keine 2 Spalten sind gleich‘):
  printf(‘ ‘):
  printf(‘  i            t1                  t2                  t3
      ‘||
       ‘      t4                   t5                  t6\n‘):

  for i from 1 to n do
   Digits:=i:
   t1 := 1/(3+sqrt(10.0))^4;
   t2 := (3-sqrt(10.0))^4;
   t3 := (19-6*sqrt(10.0))^2;
   t4 := 721-228*sqrt(10.0);
   t5 := 1/(19+6*sqrt(10.0))^2;
   t6 := 1/(721+228*sqrt(10.0));
   printf(‘ %2d   %19.16f %19.16f %19.16f %20.16f %19.16f %19.16f\n‘,
          i,t1,t2,t3,t4,t5,t6):
  end do:

  Digits:=30:
  evalf(1/(3+sqrt(10))^4);
  Digits:=10:
```

Tabelle mit Genauigkeiten `Digits=1..16`, keine 2 Spalten sind gleich
(Format der Zahlendarstellung veraendert)

i	t1	t2	t3	t4	t5	t6
1	8.000000000000E-4	0.000000000000E+0	0.000000000000E+0	1.000000000000E+2	6.000000000000E-4	1.000000000000E-3
2	6.800000000000E-4	1.600000000000E-3	0.000000000000E+0	-2.000000000000E+1	6.900000000000E-4	6.700000000000E-4
3	6.940000000000E-4	6.550000000000E-4	0.000000000000E+0	1.000000000000E+0	6.930000000000E-4	6.940000000000E-4
4	6.936000000000E-4	6.887000000000E-4	9.000000000000E-4	1.000000000000E-1	6.936000000000E-4	6.935000000000E-4
5	6.934700000000E-4	6.938600000000E-4	6.760000000000E-4	0.000000000000E+0	6.934700000000E-4	6.934800000000E-4
6	6.934810000000E-4	6.935220000000E-4	6.916900000000E-4	0.000000000000E+0	6.934800000000E-4	6.934810000000E-4
7	6.934815000000E-4	6.934874000000E-4	6.932689000000E-4	6.000000000000E-4	6.934815000000E-4	6.934818000000E-4
8	6.934815900000E-4	6.934822900000E-4	6.934795600000E-4	6.800000000000E-4	6.934816100000E-4	6.934816100000E-4
9	6.934816100000E-4	6.934816070000E-4	6.934795560000E-4	6.940000000000E-4	6.934816080000E-4	6.934816080000E-4
10	6.934816096000E-4	6.934816066000E-4	6.934816627000E-4	6.935000000000E-4	6.934816096000E-4	6.934816098000E-4
11	6.934816095000E-4	6.934816100500E-4	6.934816100500E-4	6.934700000000E-4	6.934816095100E-4	6.934816095200E-4
12	6.934816095120E-4	6.934816095400E-4	6.934816100540E-4	6.934810000000E-4	6.934816095130E-4	6.934816095120E-4
13	6.934816095125E-4	6.934816095058E-4	6.934816095268E-4	6.934817000000E-4	6.934816095123E-4	6.934816095125E-4
14	6.934816095123E-4	6.934816095127E-4	6.934816095268E-4	6.934816000000E-4	6.934816095123E-4	6.934816095123E-4
15	6.934816095123E-4	6.934816095123E-4	6.934816095110E-4	6.934816090000E-4	6.934816095123E-4	6.934816095123E-4
16	6.934816095123E-4	6.934816095123E-4	6.934816095126E-4	6.934816096000E-4	6.934816095123E-4	6.934816095123E-4

6.93481609512304252271869340176E-4

3.6 Kurvendiskussion und Approximation von Ableitungen

Betrachten wir die rationale Funktion

$$f(x) \;=\; \frac{4970x - 4923}{4970x^2 - 9799x + 4830} \tag{3.25}$$

mit der Nullstelle $x_1 = 4923/4970 = 0.9905...$ und den nahe beieinanderliegenden Polstellen $p_1 = 0.9857142823...$, $p_2 = 0.9859154963...$.
Zunächst untersuchen wir den Funktionsgrafen, den wir z. B. mit dem CAS ***Mathematica*** oder dem interaktiven Plotterprogramm **GNUPLOT** erzeugen können. Eine grobe Grafik mittels

```
      Plot [f[x],{x,0,1.2}]
bzw.  plot [0:1.2] [-150:75] f(x)
```

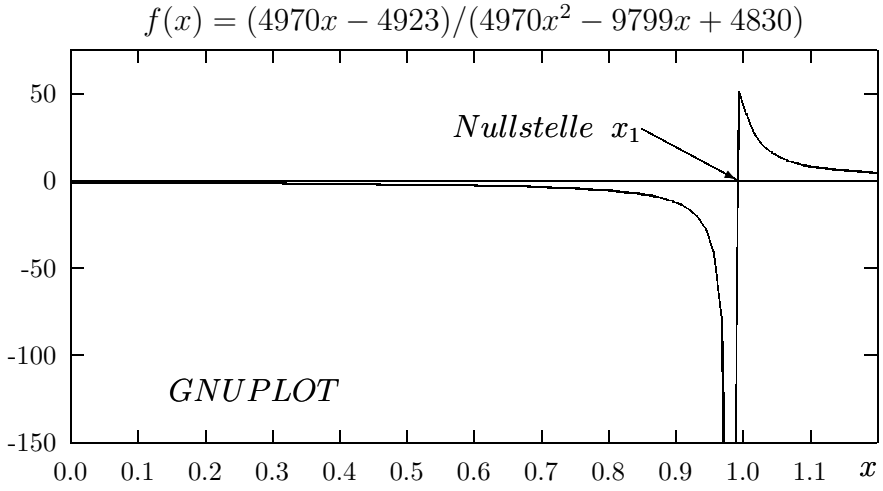
lässt zwar die Nullstelle gut erkennen, aber die Polstellen fehlen.
Geht man mittels "Fenstertechnik" langsam in den interessanten Bereich, z. B. mit

```
      Plot [f[x],{x,0.985,0.987}]
bzw.  plot [0.9854:0.9864] [-3e6:3e6] f(x)
```

so werden die Sprungstellen (bei eingezeichneten Asymptoten) sichtbar. Aber über diese Stellen mit ihren beidseitig betragsgroßen Werten wird interpoliert. Die Nullstelle ist nun rechts außerhalb des Fensters. Im letzten Schritt

```
plot [0.98565:0.98600] [-3e6:3e6] f(x)
```

sind die Asymptoten ebenfalls zusätzlich eingezeichnet worden.



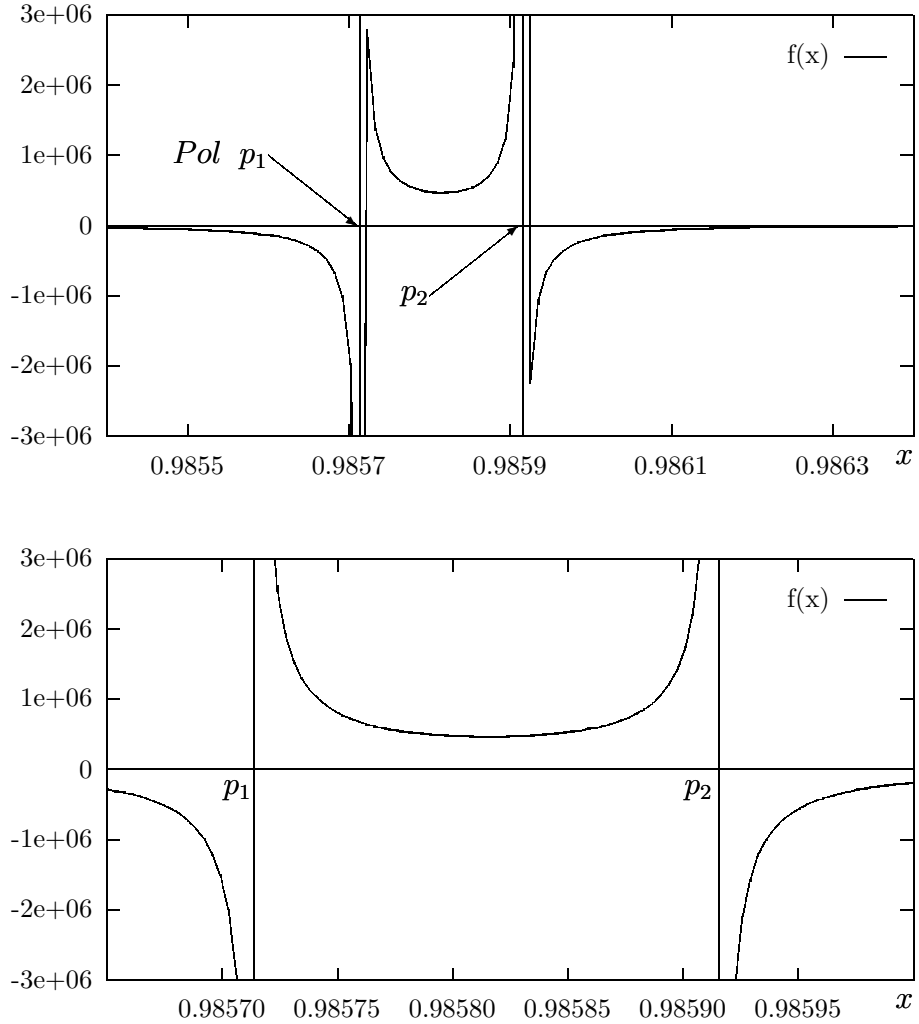


Abb. 3.7 Grafik der Funktion $f(x)$ in verschiedenen Fenstern.

Als nächstes interessieren wir uns für die näherungsweise Berechnung und Darstellung der zweiten Ableitung von $f(x)$. Es gilt

$$\begin{aligned} &f(1) = 47, \quad f''(1) = 94, \quad f''''(1) = -27851401752, \\ &\Delta_h^2 f(x) = \frac{1}{h^2}(f(x+h) - 2f(x) + f(x-h)), \\ &\text{zentraler Differenzenquotient 2. Ordnung,} \\ &\lim_{h \to 0} \Delta_h^2 f(x) = f''(x). \end{aligned} \tag{3.26}$$

Nun untersuchen wir die grafische Darstellung von $\Delta_h^2 f(x)$ als Funktion von h bei beliebigen aber festen x.
Bei der Berechnung der notwendigen Differenzen in der GP-Arithmetik mit der Mantissenlänge t weiß man, dass die Stellenauslöschung bei $h \approx \sqrt{10^{-t}}$ beginnt. Dies

steht dem Wunsch entgegen, mit möglichst sehr kleiner Schrittweite h eine gute Genauigkeit der Approximation der 2. Ableitung zu erreichen. Diese **Zwickmühle** ist in [34] für die erste Ableitung dargestellt worden. In unserem Fall stehen sich

- der relative Diskretisierungsfehler $\dfrac{h^2 f''''}{12 f''}$

- und der Auslöschungsfehler $\varepsilon \dfrac{3f}{h^2 f''}, \quad \varepsilon = 10^{-t},$

gegenüber.

Vorausgesetzt, man kann $f(x)$ mit einem relativen Fehler nahe der Mantissengenauigkeit 10^{-t} berechnen, dann erhalten wir für die Schrittweite h bei einem ausgewogenen Verhältnis der beiden Kontrahenten

$$\begin{aligned} h &= \sqrt[4]{\varepsilon \, \frac{48f}{f''''}}, \\ h &\approx \sqrt[4]{\varepsilon \; 10^{-7}}, \quad \text{falls} \quad x = 1. \end{aligned} \tag{3.27}$$

Grafische Darstellungen von $\Delta_h^2 f(1)$ machen den Sachverhalt anschaulich.

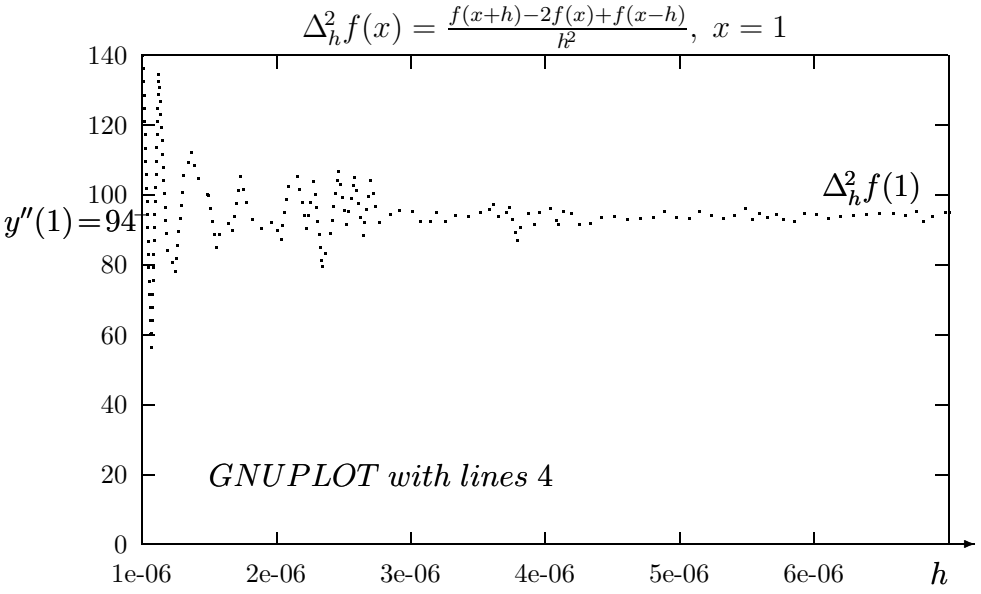


Abb. 3.8 GNUPLOT-Grafik.

Für die verschiedenen GP-Formate ergeben sich aus der folgenden Tabelle zulässige Bereiche für die Schrittweite h.

GP-Format	t	h-Bereich
extended	19...20	2E-7...1E-5
double	15...16	5E-7...1E-5
real	11...12	3E-5...4E-5
single	7...8	versagt wegen ungenauer Berechnung von $f(x)$

Tab. 3.11 Zulässige Schrittweiten h für Approximation der 2. Ableitung.

Jetzt wird die Punktfolge $(h, \Delta_h^2 f(1))$ für die zweite Ableitung mit einem TP-Programm erzeugt und als Text-File "cg23.dat" bereitgestellt. Dieses Datenfile kann man in GNUPLOT grafisch darstellen und in analoger Weise auch als TEX-File "cg23.tex" aufbereiten, um es wie hier in einen Text (Artikel) als Input-File einzubinden.

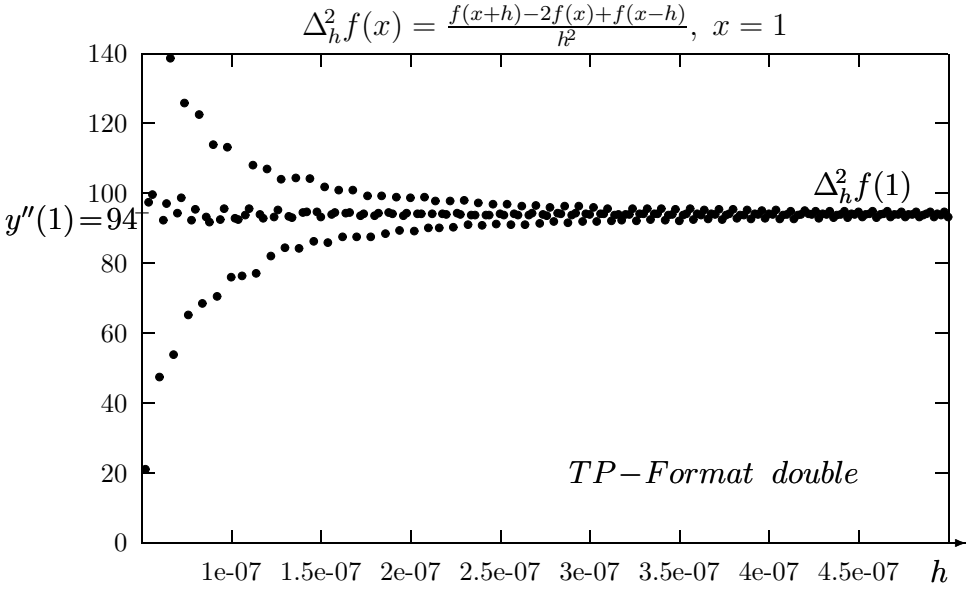


Abb. 3.9 GNUPLOT-Grafik von TP-Datei.

Für die folgende Abb. 3.7 notieren wir die GNUPLOT-Kommandos.

```
set terminal latex
set output "cg23.tex"
set size 1.0, 1.0
set title "$\Delta^{\!2}_{h}f(x)=\frac{f(x+h)-2f(x)+f(x-h)}{h^2},\
           x=1 $"
set nokey
set arrow from 0.0000002, 0 to 0.000000205, 0
```

```
set label "$h$" at 0.0000002, -8 right
set format x "%g"
set xtics 0.00000002, 0.00000004, 0.00000018
set label "$\Delta^{\!2}_{h}f(1)$" at 0.00000019, 102 right
set label "$y''(1)\!=\!94$" at 0.0000000192, 92 right
set label "--" at 0.0000000215, 94 right
set label "$TP\!-\!Format~~extended$" at 0.00000018, 20 right
f(x) = (4970*x-4923)/(4970*x**2-9799*x+4830)
plot [0.00000002:0.00000020] [0:140] "cg23.dat" with points 1 10
```

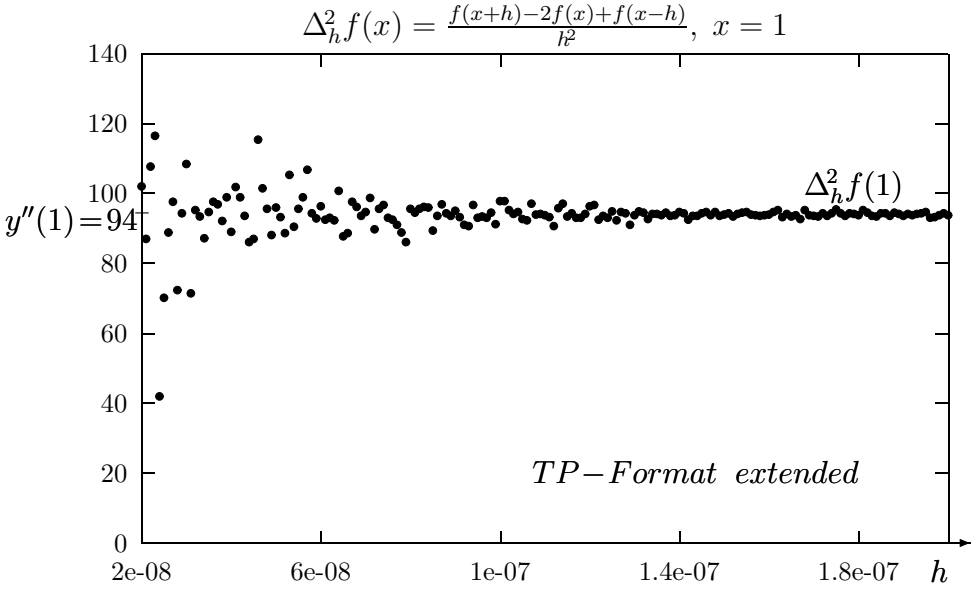


Abb. 3.10 GNUPLOT-Grafik von TP-Datei.

Rechnungen in Maple

Grafische Darstellung von Funktionen mit Unstetigkeitsstellen

Definition einer rationalen Funktion mit Nullstellen und Polstellen

```
> Digits:=10:
  x:='x':
  f:=x->(4970*x-4923)/(4970*x^2-9799*x+4830);       # Funktion
  f(x);                                             # Ausdruck, Formel
  f(1);
```

$$f := x \rightarrow \frac{4970\,x - 4923}{4970\,x^2 - 9799\,x + 4830}$$

$$\frac{4970\,x - 4923}{4970\,x^2 - 9799\,x + 4830}$$

$$47$$

```
> ‘Nullstelle‘;
  ns := solve(f(x));
  evalf(ns);
  ‘Polstellen‘;
  ps := [solve(4970*x^2-9799*x+4830)];
  evalf(ps);
```

$$Nullstelle$$
$$ns := \frac{4923}{4970}$$
$$0.9905432596$$
$$Polstellen$$
$$ps := \left[\frac{70}{71}, \frac{69}{70}\right]$$
$$[0.9859154930, 0.9857142857]$$

Ableitungen und Ableitungswerte

```
> Diff(f(x),x):
  diff(f(x),x):
  %%=%;

  D(f);       # = unapply(diff(f(x),x),x);
  D(f)(x);    # = diff(f(x),x);
  D(D(f));    # = (D@@2)(f);
```

$$\frac{d}{dx}\left(\frac{4970\,x-4923}{4970\,x^2-9799\,x+4830}\right) =$$
$$\frac{4970}{4970\,x^2-9799\,x+4830} - \frac{(4970\,x-4923)\,(9940\,x-9799)}{(4970\,x^2-9799\,x+4830)^2}$$
$$x \to \frac{4970}{4970\,x^2-9799\,x+4830} - \frac{(4970\,x-4923)\,(9940\,x-9799)}{(4970\,x^2-9799\,x+4830)^2}$$
$$\frac{4970}{4970\,x^2-9799\,x+4830} - \frac{(4970\,x-4923)\,(9940\,x-9799)}{(4970\,x^2-9799\,x+4830)^2}$$
$$x \to -\frac{9940\,(9940\,x-9799)}{(4970\,x^2-9799\,x+4830)^2} +$$
$$\frac{2\,(4970\,x-4923)\,(9940\,x-9799)^2}{(4970\,x^2-9799\,x+4830)^3} \frac{9940\,(4970\,x-4923)}{(4970\,x^2-9799\,x+4830)^2}$$

```
> f(1);
  D(f)(1);
  (D@@2)(f)(1);
  (D@@3)(f)(1);
  (D@@4)(f)(1);
```

47
-1657
94
49371978
-27851401752

```
> Int(f(x),x):
  int(f(x),x):
  %%=%;
```

$$\int \frac{4970\,x - 4923}{4970\,x^2 - 9799\,x + 4830}\,dx = -23\,\ln(71\,x - 70) + 24\,\ln(70\,x - 69)$$

Test zu Ableitungen
6 Funktionen

```
> f1:=x->x^3+x+1;
  f1(x);
```

$$f1 := x \to x^3 + x + 1$$
$$x^3 + x + 1$$

```
> f2:=unapply(x^3+x+1,x);
  f2(x);
```

$$f2 := x \to x^3 + x + 1$$
$$x^3 + x + 1$$

```
> f3:=proc(x) x^3+x+1 end;
  f3(x);
```

$$f3 := \mathbf{proc}(x)\, x^3 + x + 1\, \mathbf{end\ proc}$$
$$x^3 + x + 1$$

```
> f4:=tan;
  f4(x);
```

$$f4 := \tan$$
$$\tan(x)$$

```
> f5:=x->tan(x);
  f5(x);
```

$$f5 := \tan$$
$$\tan(x)$$

```
> tan(x);
```

$$\tan(x)$$

Analog fuer alle 6 Funktionen: 1. Variante fehlerhaft
4 zulaessige Varianten

```
> Diff(f1(x),x)=diff(f1(x),x);

  # Fehlerhaft
  f1s:=x->diff(f1(x),x);
  f1s(2);
```

$$\frac{d}{dx}\left(x^3+x+1\right)=3\,x^2+1$$

$$f1s := x \to \mathrm{diff}(\mathrm{f1}(x),\,x)$$

```
Error, (in f1s) wrong number (or type) of parameters in function diff

> # 4 zulaessige Varianten, Ableitung als Funktion
  f1s1:=unapply(diff(f1(x),x),x);
  f1s1(2);

  D(f1);
  D(f1)(x);
  D(f1)(2);
  subs(x=2,D(f1)(x));
  eval(D(f1)(x),x=2);

  f1s2:=D(f1);
  f1s2(2);

  f1s3:=x->D(f1)(x);
  f1s3(2);

  f1s4:=proc(x) D(f1)(x) end;  # genauso mit anderen Funktionen machbar
  f1s4(2);
```

$$f1s1 := x \to 3\,x^2+1$$
$$13$$
$$x \to 3\,x^2+1$$
$$3\,x^2+1$$
$$13$$
$$13$$
$$13$$
$$f1s2 := x \to 3\,x^2+1$$
$$13$$
$$f1s3 := x \to 3\,x^2+1$$
$$13$$
$$f1s4 := \mathbf{proc}(x)\ \mathrm{D}(f1)(x)\ \mathbf{end\ proc}$$
$$13$$

Funktion $f1$ und ihre Ableitung $f1s1,\ f1s2,\ f1s3,\ f1s4$ in Prozeduren global und/oder lokal

```
> verf1:=proc(x0)
        local x;
        global f1,f1s1;        # f1s1,f1s2,f1s3,f1s4
   x:=x0+2;
   evalf(f1(x)*f1s1(x));       # 11*13=143
  end;

  verf1(0);
```

$$verf1 := \mathbf{proc}(x0)\ \ldots\ \mathbf{end\ proc}$$
$$143.$$

```
> verf2:=proc(x0::numeric,fs::procedure)
        local x;
        global f1;
   x:=x0+2;
   evalf(f1(x)*fs(x));  # 11*13=143
  end;

  verf2(0,f1s1),verf2(0,f1s2),verf2(0,f1s3),verf2(0,f1s4);
```

$$verf2 := \mathbf{proc}(x0 :: numeric,\ fs :: procedure)\ \ldots\ \mathbf{end\ proc}$$
$$143., 143., 143., 143.$$

```
> verf3:=proc(x0::numeric,f::procedure,fs::procedure)
        local x;
   x:=x0+2;
   evalf(f(x)*fs(x));  # 11*13=143
  end;

  verf3(0,f1,f1s1),verf3(0,f1,f1s2),verf3(0,f1,f1s3),verf3(0,f1,f1s4);
```

$$verf3 := \mathbf{proc}(x0 :: numeric,\ f :: procedure,\ fs ::\ procedure)\ \ldots\ \mathbf{end\ proc}$$
$$143., 143., 143., 143.$$

```
> verf4:=proc(x0::numeric,f::procedure)
        local x,fss;
   fss:=D(f);
   x:=x0+2;
   evalf(f(x)*fss(x));  # 11*13=143
  end;

  verf4(0,f1);
```

$$verf4 := \mathbf{proc}(x0 :: numeric,\ f :: procedure)\ \ldots\ \mathbf{end\ proc}$$
$$143.$$

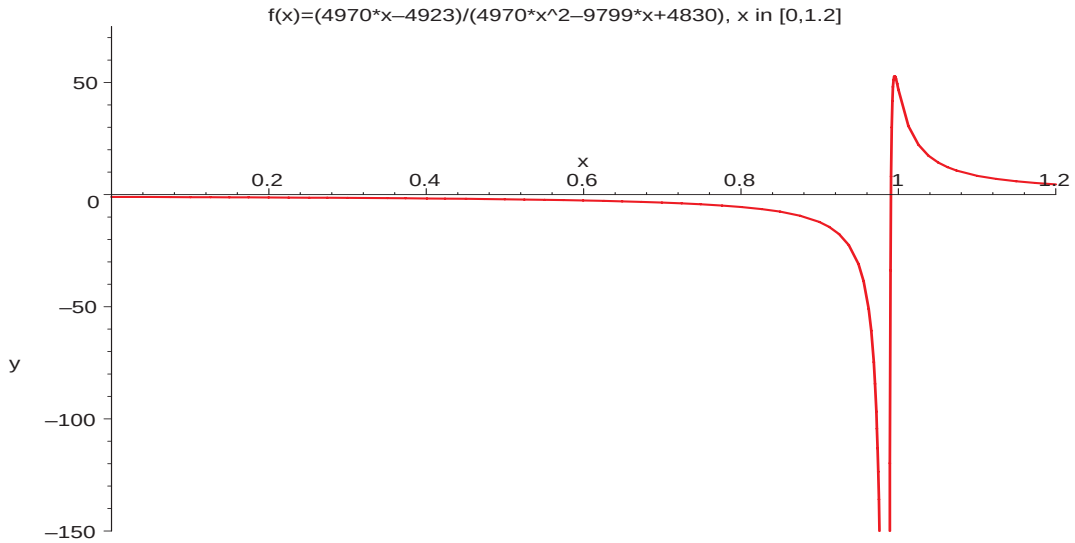
Rueckkehr zur Berechnung der 2. Ableitung
Grobe Grafik ist problematisch, deshalb sollte man in das kritische Gebiet mit NS und PS hineinschauen (zoomen).

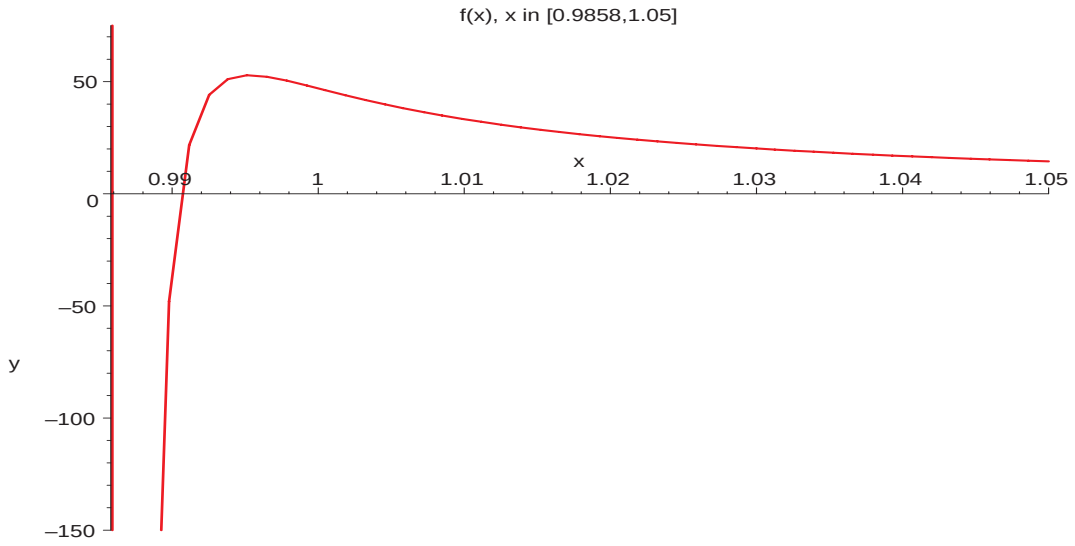
```
> plot(f(x),x=0..1.2,y=-150..75,thickness=3,
      title=‘f(x)=(4970*x-4923)/(4970*x^2-9799*x+4830), x in [0,1.2]‘);

> plot(f(x),x=0.9858..1.05,y=-150..75,thickness=3,
      title=‘f(x), x in [0.9858,1.05]‘);

> plot(f(x),x=0.9854..0.9864,y=-3e6..3e6,thickness=3,
      title=‘f(x)=(4970*x-4923)/(4970*x^2-9799*x+4830),
             x in [0.9854,0.9864]‘);

> plot(f(x),x=0.98565..0.98600,y=-3e6..3e6,thickness=3,
      title=‘f(x), x in [0.98565,0.98600]‘);
```





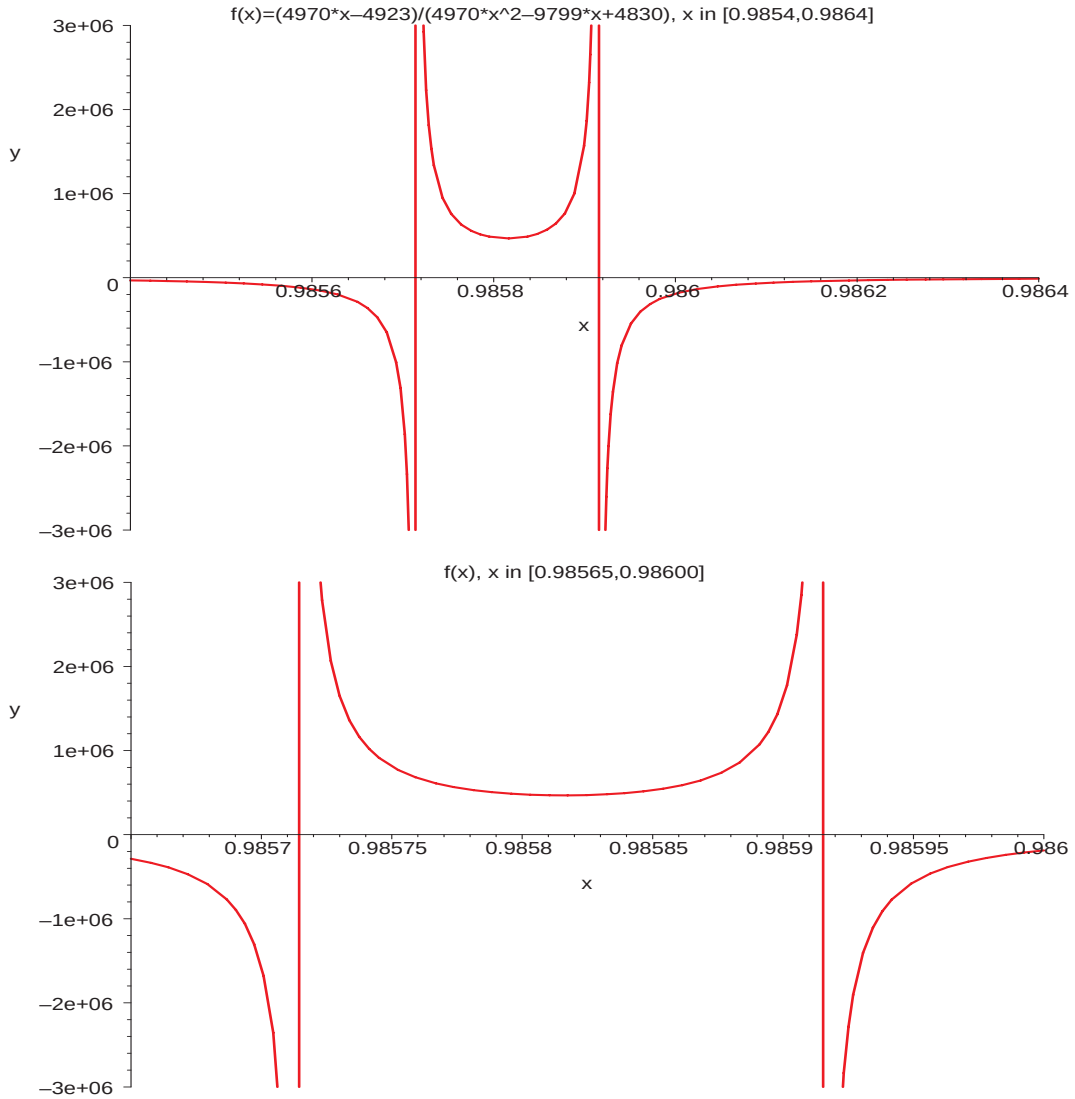


Abb. 3.11 Funktion $f(x) = \dfrac{4970x - 4923}{4970x^2 - 9799x + 4830}$ mit x aus 4 Bereichen

2. Aleitung $f''(x)$ und Approximation mit zentralem Differenzenquotient $d2(x,h) = \Delta_h^2 f(x),\ x = 1,\ h > 0$

```
> (D@@2)(f);
  (D@@2)(f)(1);
```

$$x \to -\frac{9940\,(9940\,x - 9799)}{(4970\,x^2 - 9799\,x + 4830)^2} + \frac{2\,(4970\,x - 4923)\,(9940\,x - 9799)^2}{(4970\,x^2 - 9799\,x + 4830)^3} - \frac{9940\,(4970\,x - 4923)}{(4970\,x^2 - 9799\,x + 4830)^2}$$

$$94$$

```
> Digits:=16:
  d2:=(x,h)->(f(x+h)-2*f(x)+f(x-h))/h^2;   # f"(1)-d2(1,h)=-h^2/12 f^(4)
  d2(1,1e-5);
```

$$d2 := (x,\, h) \rightarrow \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

$$93.76790000000000$$

```
> #  Test von Digits=5,6,...  bei fester Schrittweite h=1e-5>0
  n:=30:
  i:='i':
  printf(` Digits   Approximation fuer f"(1)\n`):
  for i from 5 by 1 to n do
   Digits:=i:
   printf(` %2d     %25.16f \n`,i,d2(1,1e-5)):
  end do:
```

```
  Digits   Approximation fuer f"(1)
    5             0.0000000000000000
    6             0.0000000000000000
    7             0.0000000000000000
    8        460000.0000000000000000
    9        467000.0000000000000000
   10        467300.0000000000000000
   11         -2730.0000000000000000
   12         -2726.0000000000000000
   13            93.8000000000000000
   14            93.7700000000000000
   15            93.7680000000000000
   16            93.7679000000000000
   17            93.7679100000000000
   18            93.7679050000000000
   19            93.7679047000000000
   20            93.7679047600000000
   21            93.7679047550000000
   22            93.7679047547000000
   23            93.7679047546500000
   24            93.7679047546510000
   25            93.7679047546509000
   26            93.7679047546509500
   27            93.7679047546509550
   28            93.7679047546509551
   29            93.7679047546509551
   30            93.7679047546509551
```

Grafische Darstellung der Approximation mit wachsender Genauigkeit

```
> h:='h':  y:='y':
  p2:=plot([[5e-7,94],[5e-6,94]],color=black,thickness=3):
  p3:=textplot([[5e-7,94,`94   `],[5e-7,97,`_`]],align=LEFT):

  Digits:=10:
  p11:=plot(d2(1,h),h=5e-7..5e-6,y=0..140,color=blue):
  plots[display]([p2,p11,p3],labels=[`h`,``],title=
   `Approximation von f"(1)=94 mit zentr. DQ d2(1,h) bei h->0,D=10..14`);

  Digits:=15:
  p12:=plot(d2(1,h),h=5e-7..5e-6,y=0..140,color=blue):
  plots[display]([p2,p12,p3],labels=[`h`,``],title=
   `Approximation von f"(1)=94 mit zentr. DQ d2(1,h) bei h->0,D=15`);

  Digits:=16:
  p13:=plot(d2(1,h),h=5e-7..5e-6,y=0..140,color=blue):
  plots[display]([p2,p13,p3],labels=[`h`,``],title=
   `Approximation von f"(1)=94 mit zentr. DQ d2(1,h) bei h->0,D=16`);

  Digits:=17:
  p14:=plot(d2(1,h),h=5e-7..5e-6,y=0..140,color=blue):
  plots[display]([p2,p14,p3],labels=[`h`,``],title=
   `Approximation von f"(1)=94 mit zentr. DQ d2(1,h) bei h->0,D=17`);

  Digits:=18:
  p14:=plot(d2(1,h),h=5e-7..5e-6,y=0..140,color=blue):
  plots[display]([p2,p14,p3],labels=[`h`,``],title=
   `Approximation von f"(1)=94 mit zentr. DQ d2(1,h) bei h->0,D=18..19`);
```

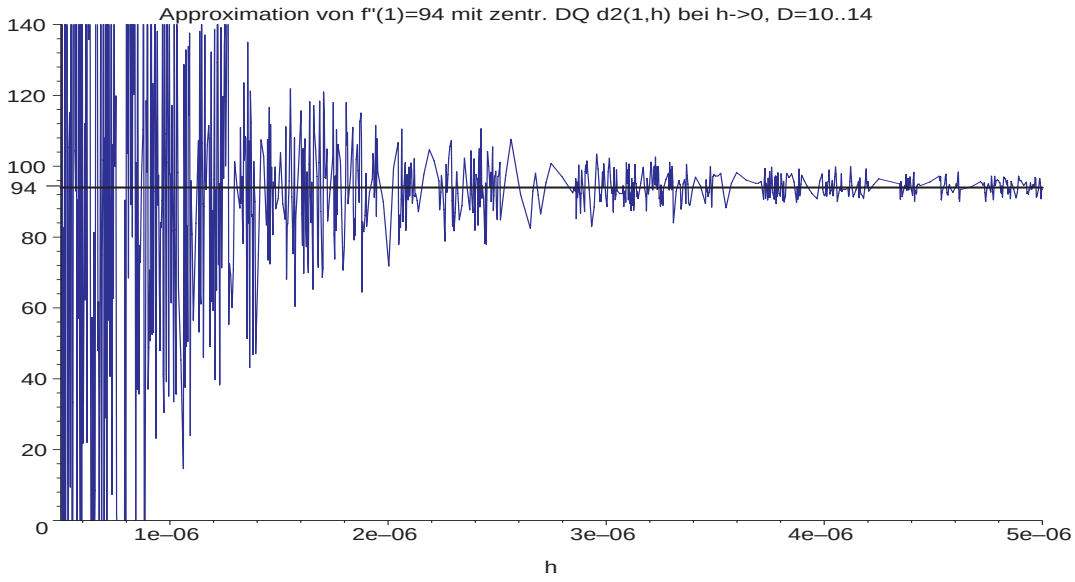


Abb. 3.12 Approximation von $f''(1) = 94$ mit zentr. DQ, Digits=10..14

$$d2(1,h) = \Delta_h^2 f(x) = \frac{1}{h^2}(f(x+h) - 2f(x) + f(x-h)),\ h \in [h_0, h_1],\ x = 1$$

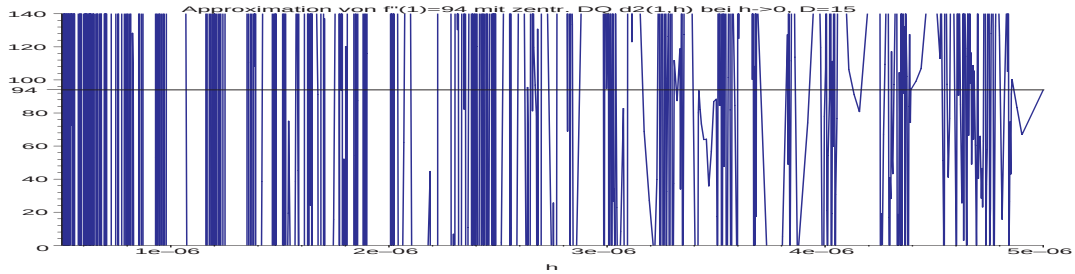


Abb. 3.13 Approximation von $f''(1) = 94$ mit zentr. DQ, Digits=15

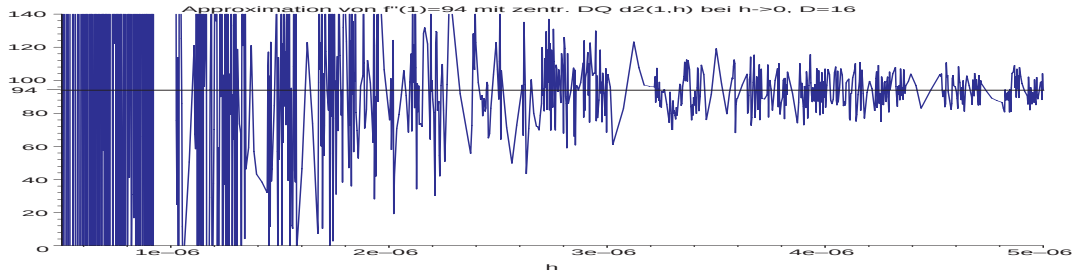


Abb. 3.14 Approximation von $f''(1) = 94$ mit zentr. DQ, Digits=16

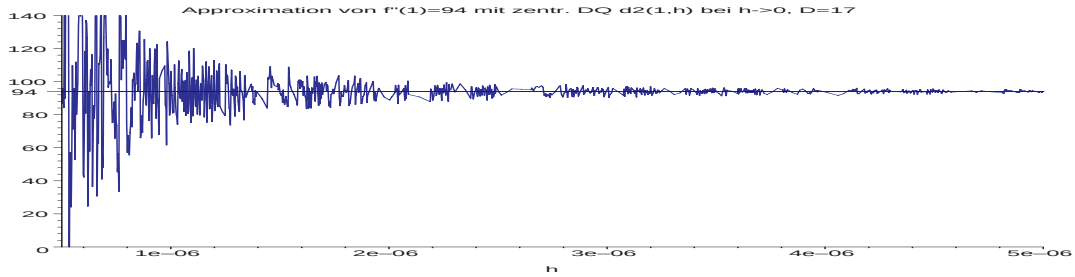


Abb. 3.15 Approximation von $f''(1) = 94$ mit zentr. DQ, Digits=17

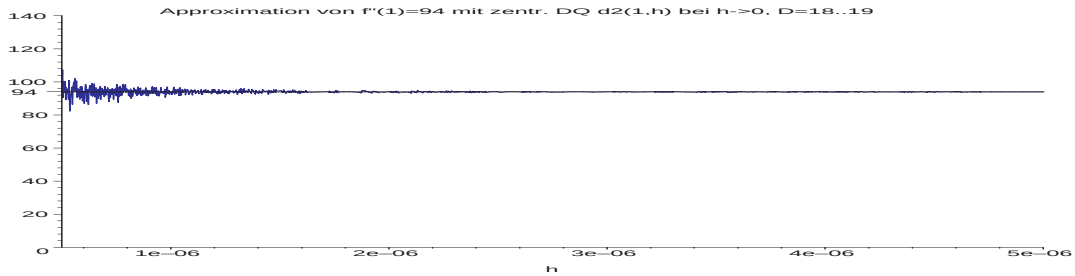


Abb. 3.16 Approximation von $f''(1) = 94$ mit zentr. DQ, Digits=18..19

Es scheint so, dass für numerische Auswertungen in Maple die standardmäßig eingestellte GP-Arithmetik mit `Digits=10` (entspricht dem Format *real*) relativ gut funktioniert. Zumindest liefern in den Rechnungen die Einstellungen `Digits=11,12,13,14` keine besseren Ergebnisse, unverständlicherweise manchmal sogar schlechtere, wie mit `Digits=15`. Erst mit dem Übergang zum Format *double* (`Digits=16`) sind Verbesserungen zu erwarten.

3.7 Computergenauigkeit und PC-Null

Für manche Aufgabenstellungen ist es wünschenswert oder sogar notwendig, sich einige Informationen über die Computergenauigkeit in der gegebenen GP-Arithmetik zu verschaffen. Für das Format *real* in Turbo Pascal gilt z. B.

(1) z = kleinste positive reelle Zahl = 2.9E-39,

(2) aus $1 + x \neq 1$ und $1 + x/2 = 1$, ergibt sich x =1.8E-12,

(3) die Anzahl der gültigen dezimalen Mantissenstellen beträgt 11...12.

Die Bestimmung der Zahl z kann mittels einer Schleifenanweisung erfolgen.

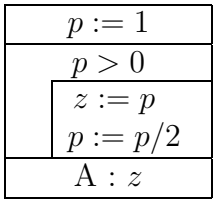


Abb. 3.17 Struktogrammteil für $z = \min\limits_{p>0} p$.

Natürlich ist auch ein anderer Quotient als 2 für die Reduktion möglich.
In TP kann man dieses Konstrukt durch eine Rekursivität in zwei Formen darstellen.

```
type float = real;
function REKURSION1(P:float):float;
var Z:float;
begin
  Z:=P;
  P:=P/2;
  if P>0 then REKURSION1:=REKURSION1(P)
         else REKURSION1:=Z;
end;

function REKURSION2(P:float):float;
var Z:float;
begin
  Z:=P;
  if P/2>0 then REKURSION2:=REKURSION2(P/2)
           else REKURSION2:=Z;
end;
```

Mit dem Aufruf und der Ausgabe `writeln(REKURSIONi(1)); i=1,2` erhalten wir den Wert 2.9387358771E-39.
Anders verhält sich die Situation in den GP-Formaten *single*, *real*, *double* oder *extended* mit der Compilerdirektive $**N**+ (Numerik-Koprozessor).

Hier muss man beachten, dass es einen sogenannten **d-Bereich** gibt, der von der kleinsten positiven normierten reellen Zahl (1. Ziffer der Mantisse $\neq 0$) bis zur kleinsten positiven intern darstellbaren Zahl reicht. Es kommt also die Mantissenlänge im Exponenten hinzu.

GP-Format	t	d-Bereich
single	7...8	1.40E-45.....1.18E-38
real	11...12	.....2.93E-39
		kein d-Bereich
double	15...16	4.94E-324.....2.23E-308
extended	19...20	1.90E-4951....3.30E-4932

Tab. 3.12 d-Bereiche der GP-Formate in Turbo Pascal.

Werte von Ausdrücken sind für alle 4 Formate bis zur Größenordnung *extended* möglich. Variablen vom Typ *single*, *double* bzw. *extended* können Werte aus dem d-Bereich aufnehmen und werden erst zu Null, wenn dieser Wertebereich unterschritten wird. Mit dem Funktionsaufruf `REKURSION1(1)` erhalten wir folgende kleinste positive Werte z.

GP-Format	Schrittzahl	z
single	150	1.4E-45
real	129	2.9E-39
double	1075	4.9E-324
extended	16446	1.9E-4951

Tab. 3.13 $z = \min\limits_{p>0} p$ mittels `REKURSION1(1)` und $N+.

Weitere Besonderheiten sind:

- Beim GP-Format *extended* werden die Werte aus dem d-Bereich als 0 ausgegeben.
- Im d-Bereich "verlieren" die Variablen gültige Mantissenstellen. Deshalb sollten Rechnungen dort möglichst unterbleiben.

Beim Funktionsaufruf `REKURSION2(1)` kann der Ausdruck $p/2$ im d-Bereich liegen. Beim wiederholten Aufruf von `REKURSION2(p/2)` wird eine lokale Hilfsvariable h im Kellerspeicher bereitgestellt, die erstens denselben Typ wie p hat und zweitens den Wert $p/2$ aufnimmt. Damit wird $h = 0$, falls $p/2$ erstmalig kleiner als der d-Bereich ist, und `REKURSION2` kehrt mit dem Wert 0 zurück. Es wird also im Vergleich zu `REKURSION1(1)` ein Schritt zuviel ausgeführt.

Rechnungen in Maple

Computernull ohne/mit Rekursion

Mantisse und Exponent von GPZ sind integer-Groessen (10..11 Ziffern),
Ueberlauf dieser Stellen fuehrt zu einer Fehlermeldung.

```
> Digits:=10:
  Reku1:=proc(p,nen)
         local q,z;
   q:=p;
   if q<=0 then RETURN(0)
           else
    while q>0 do
     z:=q;
     print(z);
     q:=q/nen;
    od;
    RETURN(z);
  fi;
  end:
```

Grob- und Feinrechnung mit verschiedenen Teilern

```
> klg:=Reku1(1.0,1e210000000);            # Grobrechnung
  klf:=Reku1(1e-2147000000,1e100000);     # Feinrechnung
                           # 1e-2147400000 ~ kleinste positive Zahl
```

$$1.0$$
$$0.1000000000\,10^{-209999999}$$
$$0.1000000000\,10^{-419999999}$$
$$0.1000000000\,10^{-629999999}$$
$$0.1000000000\,10^{-839999999}$$
$$0.1000000000\,10^{-1049999999}$$
$$0.1000000000\,10^{-1259999999}$$
$$0.1000000000\,10^{-1469999999}$$
$$0.1000000000\,10^{-1679999999}$$
$$0.1000000000\,10^{-1889999999}$$
$$0.1000000000\,10^{-2099999999}$$
$$klg := 0.1000000000\,10^{-2099999999}$$

$$0.1\,10^{-2146999999}$$
$$0.1000000000\,10^{-2147099999}$$
$$0.1000000000\,10^{-2147199999}$$
$$0.1000000000\,10^{-2147299999}$$
$$0.1000000000\,10^{-2147399999}$$
$$klf := 0.1000000000\,10^{-2147399999}$$

Auf diese Weise kommt ganz in die Nähe der kleinsten positiven Zahl in Maple.

3.8 Dateiarbeit, Geometrie und Figuren

3.8.1 Dateiarbeit beim Speichern von Figuren als File

In [10] sind einige grafische Ausgabemöglichkeiten unter Maple sowie in Verbindung mit LaTeX erläutert worden. Inzwischen gibt es in Maple auch die Möglichkeit, eine auf dem Bildschirm erzeugte Grafik durch eine implementierte Menüfunktion zu exportieren. Außerdem hat sich bei Maple → LaTeX mit dem Export von Bildern etwas verändert. Wir fassen die wichtigsten Varianten des Grafik-Exports noch einmal zusammen.

1. **Maple-Arbeitsblätter mit Grafik**

   Zunächst bringt der Druck des Arbeitsblatts den darin enthaltenen Plot ungefähr in der Größe $17 \times 17cm$ horizontal zentriert aufs Papier. Natürlich kann man den Plot auch zoomen und dann in veränderter Größe ausdrucken.

2. **Maple-Arbeitsblätter mit Grafik und ihr Export nach LaTeX**

   Der Export des Arbeitsblatts *name.mws* nach LaTeX erzeugt neben dem LaTeX-File *name.tex* für die darin enthaltenen Plots zusätzlich die entsprechenden Postscript-Dateien (*eps*-Format, Encapsulated PostScript) *name01.eps*, *name02.eps*, usw. Jede einzelne Grafik befindet sich in einer Box mit den Grenzen (0,0,287,216), also der Dimension $287 \times 216pt = 101 \times 76mm$, am linken unteren Rand der A4-Seite. Die Grafikfiles haben somit ein kleines Format, sind *portrait* und *monochrom (schwarz/weiß)*. Je nach Inhalt können diese bis mehrere Megabyte groß sein.

   Die Datei enthält u. a. die genannten Angaben

```
%!PS-Adobe-3.0 EPSF-2.0
%%Title: Maple plot
%%Creator: Maple
%%Pages:  1
%%BoundingBox: 0 0 287 216
%%DocumentNeededResources: font Helvetica
%%EndComments
20 dict begin
...
%%IncludeResource: font Helvetica
0.000000 0.000000 translate
...
showpage
grestore
end
%%EOF
```

Die Ausgabe des Grafikfiles kann erfolgen

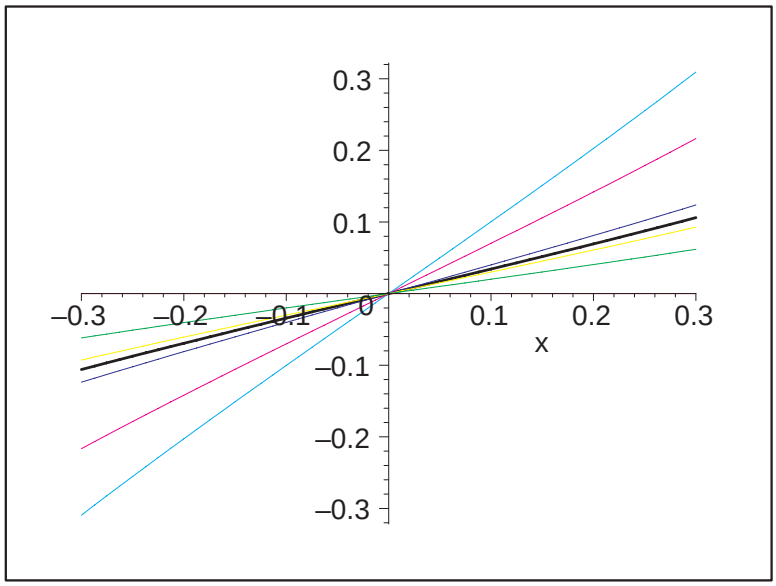
- im Rahmen von Maple-Gruppen.
  Da gibt es 3 Varianten, wie sie nachfolgend aufgeführt sind.

```
\begin{maplegroup}
 ...
 \mapleresult
 \psfig{figure=name01.eps}
 \begin{maplelatex} \mapleplot{name01.eps} \end{maplelatex}
 \begin{center} \mapleplot{name01.eps} \end{center}
  ...
\end{maplegroup}
```

  Am besten ist es, in der Maple-Gruppe das Kommando

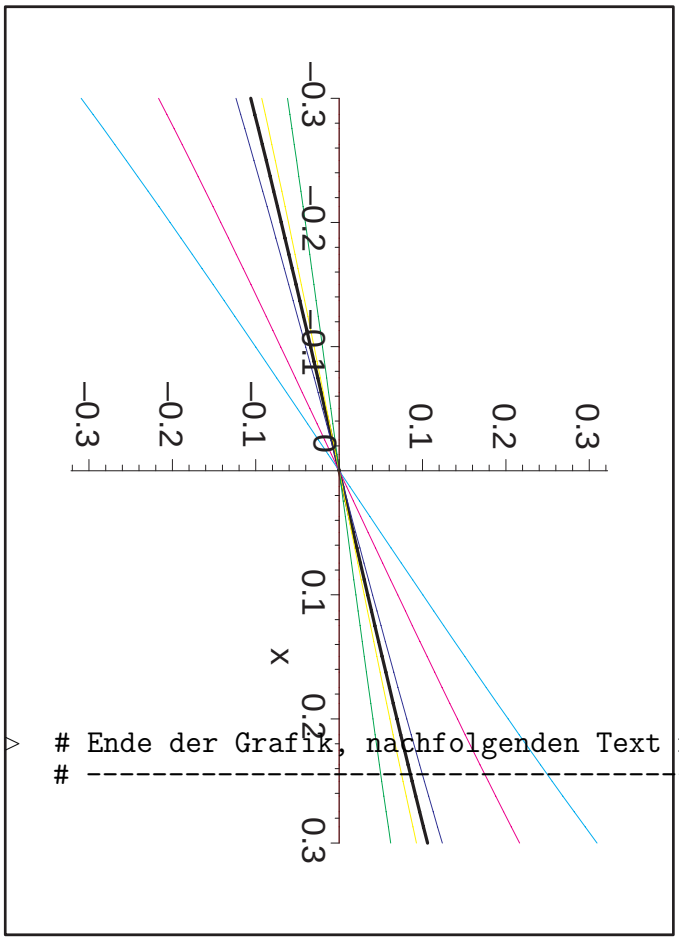
  `\psfig{figure=...}` mit dem Stil *psfig.sty* zu nehmen,

  wo die Grafik in ihrer Originalgröße in einer Box $101 \times 76mm$ und im Format *portrait* erscheint.



  Bei den anderen beiden Möglichkeiten erscheint die Grafik im *ps*-File *name01.eps* in der Ausgabe in einem berandeten Rechteck der Dimension von ca. $57 \times 85mm$ in einer Box von ca. $70 \times 97mm$, also etwas kleiner als die Originalgröße $76 \times 101mm$. Dabei ist sie noch um –90 Grad gedreht. Skalierungsangaben `width=...`, `height=...`, bzw. `angle=...` sind im Befehl `\mapleplot{name01.eps}` nicht zulässig. Weiterhin ragt der nachfolgende Text in die Grafik hinein. Damit ist eine Nachbereitung des LaTeX-Textes notwendig.

```
> plot([fl(x),fr(x)],x=Ie,thickness=2);
  p1:=plot([fl1(x),fl2(x),fl3(x),fl4(x),fl5(x),fl6(x)],x=Ie,
           thickness=2):
  p2:=plot(fr(x),x=Ie,thickness=4,color=black):
  plots[display](p1,p2);
```




```
> # Ende der Grafik, nachfolgenden Text nach unten verschieben
  # ------------------------------------------------------------
```

- im LaTeX-Text außerhalb von Maple-Text mit dem Stil *psfig.sty* oder *epsf.sty*.

  Beispielvarianten

```
% im Original ist die Grafik von der Groesse 101x76 mm
% und Portrait
\psfig{figure=name01.eps}

% normale Ansicht als Portrait mit Groessenangaben
\psfig{figure=name01.eps,width=13cm,height=10cm}

\epsfbox{name01.eps}
\epsfbox[0 0 w h]{name01.eps}
```

3. **Grafik-Export durch implementierte Menüfunktion**

   Der ausgewählte Plot wird in eine Postscript-Datei (*eps*-Format, Encapsulated PostScript) mit frei wählbarem Namen gespeichert. Die Grafik befindet sich in einer Box mit den Grenzen (72,72,540,719), also der Dimension $468 \times 647pt = 165 \times 227mm$, diese ist um 90 Grad gedreht und noch verschoben (540.000000 72.000000 translate). Damit liegt sie zentriert auf der A4-Seite.

   Die Ausgabe des Grafikfiles kann wie vorher erfolgen. Dabei ist sinvollerweise eine Drehung um –90 Grad angebracht und evtl. die Bildgrößen zu verändern (verkleinern), z. B. `\psfig{figure=exam1.eps,width=13cm,angle=-90}`. Beim Export sind auch andere Formate möglich.

4. ***interface*-Funktion für Grafik mit Ausgabeformaten**

   Wir zitieren aus [7].
   The function *interface* is provided as a unique mechanism of communication between Maple and the user interface. Specifically, this function is used to set and query all variables which affect the format of the output but do not affect the computation.
   Its arguments are specifies, i. e. for plots
   plotdevice : The name of the plotting device.
   plotoutput : Name of a file where the plot output will be stored.
   plotoptions : Contain device specific options to be passed to the device driver.
   Using the *interface* command you are able to store pictures.

   - Als übliche und auch elegante Variante empfiehlt sich somit die Verwendung des Maple-Befehls

     ```
     interface(plotdevice=..., plotoutput=..., plotoptions=... );
     ```

     Damit kann man die erzeugten Grafiken wahlweise als *ps (PostScript)-, gif (Graphics Interchange Format)-, jpg (Joint Picture Experts Group (JPEG))-* oder *pcx* (*PCPaintbrush*)-File abspeichern.

     Vorgehensweise

     Ein 3D-Plot soll als Postscript-File im Format *portrait* und *monochrom* mit gegebener Größe erstellt werden.

     ```
     > restart:
       with(plots):
       plot3d(sin(x)*exp(y),x=0..10,y=1..4);

     > interface(plotdevice=ps,
                 plotoutput=‘C:/D/Neundorf/Maple3/bild1.ps‘,
                 plotoptions=‘portrait,noborder‘);
       plot3d(sin(x)*exp(y),x=0..10,y=1..4);
     > interface(plotdevice=win);   # Standard output
     ```

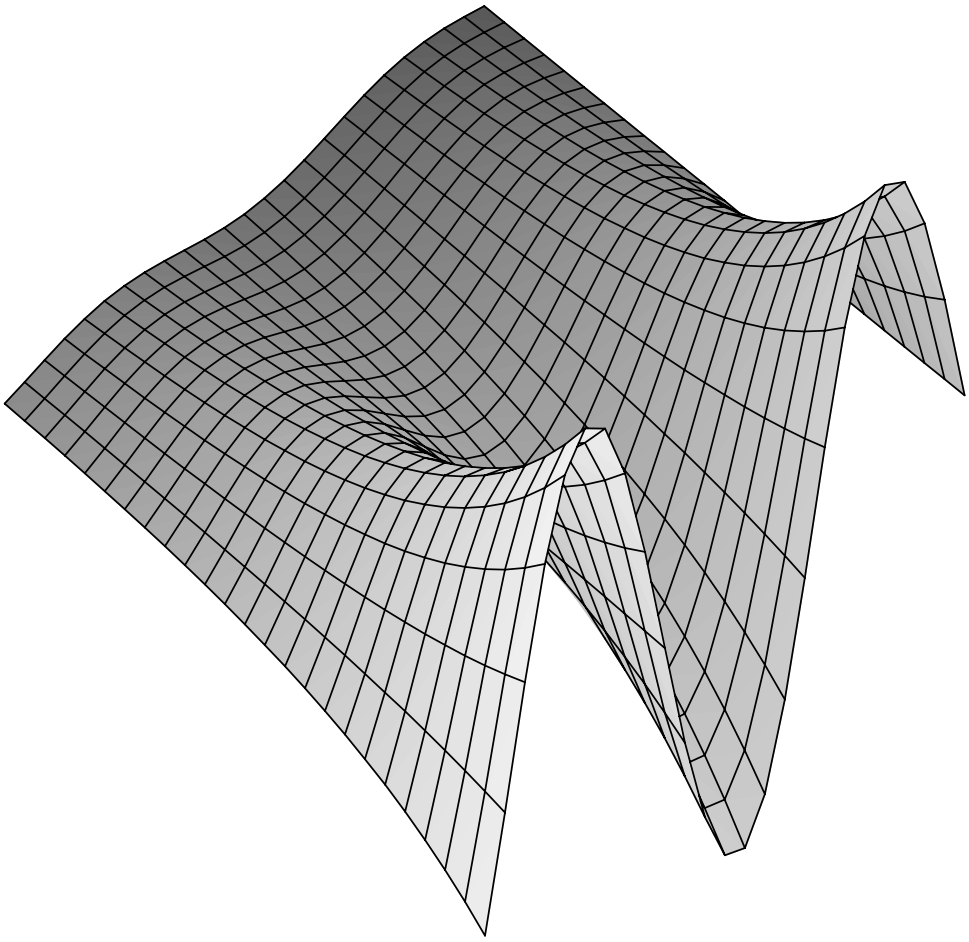


Abb. 3.18 Datei *bild1.ps*, $f(x,y) = \sin(x)e^y,\ x = 0..10, y = 1..4$.

Es ist empfehlenswert, nach der obigen Umlenkung der Ausgabe auf das File im angegebenen Verzeichnis anschließend sofort wieder die Standardausgabe auf dem Bildschirm zu aktivieren, da sonst alle nachfolgenden Grafiken ebenso unter dem gewählten Filenamen abgespeichert werden und damit die erste Datei ohne Vorwarnung überschrieben wird.

Die Angabe des Grafikfiles erfolgt vorzugsweise mit LW:/Pfad/Dateiname.

plotdevice	plotoutput File *.*	plotoptions
ps	name.ps	Standard: *landscape, s/w* möglich auch *portrait, color, width, height*
gif	name.gif	Standard: *color, portrait, transparent=false* möglich auch *transparent=true* (transparent)
jpeg	name.jpg	Standard: *color, portrait*
pcx	name.pcx	Standard: *color, portrait*

Tab. 3.14 Ausgabeformate und Eigenschaften

Betrachtet man sich die Beispielgrafik *bild1.ps* mit einem *ps*-Viewer, so erscheint diese auf einer A4-Seite mit der Bildgröße $359 \times 348pt = 126 \times 123mm$ um $(72, 72)pt$ von der linken unteren Ecke in die Mitte verschoben. Aus der Datei sind diese Angaben zu entnehmen.

```
%!PS-Adobe-3.0 EPSF-2.0
%%Title: Maple plot
%%Creator: Maple
%%Pages:  1
%%BoundingBox: 126 193 485 541
%%DocumentNeededResources: font Helvetica
%%EndComments
...
%%IncludeResource: font Helvetica
72.000000 72.000000 translate
...
```

Die Wirkung der Größenangaben in den Plotoptionen im Zusammenhang mit den entsprechen Parameterangaben in den *ps*-Files zeigen die folgenden drei Einstellungen von Plotoptionen im *interface*-Kommando.
Dabei hängen die Dimensionen der Bounding Box von den Plotoptionen ab, insbesondere von `scaling=constrained` oder `unconstrained` und später im *interface*-Befehl von `portrait, landscape, width=...`, `height=...`.

Nachfolgend sei beispielhaft angenommen, dass der Originalplot "maximal" ist, was der Einstellung mit freiem Maßstab `scaling=unconstrained` entspricht.

```
> pfad:=‘C:/D/Neundorf/Maple3/‘:
  dateiname:=‘int_02‘:
  file:=cat(pfad,dateiname,‘.ps‘):

  # Infos zum abgespeicherten ps-File (1pt=0.35146mm)

  interface(plotdevice=ps,plotoutput=file,
            plotoptions=‘portrait,noborder’);

  # ps-File/Bild hat Bounding Box [93..507]x[95..615] der
  # Dimensionen 414*560pt=145.5*196.8mm und ist verschoben
  # um (72,72)pt von linker unterer Ecke in Bildmitte

  #       %%BoundingBox: 93 95 507 655
  #       72.000000 72.000000 translate
```

```
interface(plotdevice=ps,plotoutput=file,
    plotoptions='portrait,noborder,width=468,height=648');

 # ps-File/Bild hat Bounding Box [93..507]x[95..615] der
 # Dimensionen 414*560pt=145.5*196.8mm und ist verschoben
 # um (72,72)pt von linker unterer Ecke in Bildmitte

 #       %%BoundingBox: 93 95 507 655
 #       72.000000 72.000000 translate

 interface(plotdevice=ps,plotoutput=file,
     plotoptions='portrait,noborder,width=414,height=560');

 # ps-File/Bild hat Bounding Box [103..508]x[135..619] der
 # Dimensionen 405*484pt=142.3*170.1mm und ist verschoben
 # um (99,116)pt von linker unterer Ecke in Bildmitte

 #       %%BoundingBox: 103 135 508 619
 #       99.000000 116.000000 translate

 plots[display](...);
 interface(plotdevice=win);
```

- Analog kann die Umlenkung der Ausgabe der Grafik auch im Zusammenhang mit dem Befehl

  ```
  plotsetup(devicetype, plotoutput=..., plotoptions=...)
  ```

  erfolgen. Dies funktioniert wie eine Parametereinstellung für Plots.
  Dann hat man die Kommandofolge

```
> restart:
  with(plots):

>  plot(1/2*(x-1)/(1+x^2)+1/2*arctan(x),x=-2..2);

> interface(plotdevice=win);
  plotsetup(ps,
      plotoutput='C:/D/Neundorf/Maple3/bild2.ps',
      plotoptions='portrait,noborder,width=200,height=200');
      plot(1/2*(x-1)/(1+x^2)+1/2*arctan(x),x=-2..2);

> plotsetup(default);          # Standard output
```

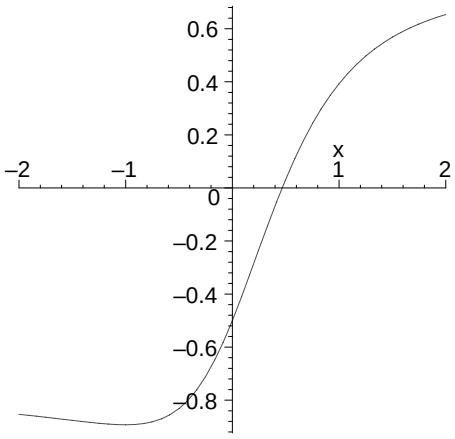


Abb. 3.19 Datei *bild2.ps*, $f(x) = \frac{1}{2}\Big(\frac{x-1}{1+x^2} + \arctan(x)\Big),\ x = -2..2.$

Aus der Datei erkennt man folgende Größenangaben.

```
%!PS-Adobe-3.0 EPSF-2.0
%%Title: Maple plot
%%Creator: Maple
%%Pages:  1
%%BoundingBox: 221 315 388 476
%%DocumentNeededResources: font Helvetica
%%EndComments
...
%%IncludeResource: font Helvetica
206.000000 296.000000 translate
...
```

An der Beispielgrafik *bild2.ps* wollen wir mit sich verändernden Angaben `width=...,height=...` die Auswirkungen auf die Größe und Verschiebung des Bildes *bild2.ps* auf einer A4-Seite kontrollieren.

`width`	`height`	`BoundingBox` in *pt*				Dimension BB in *mm*	`translate` in *pt*	
200	200	221	315	388	476	59×57	206	296
400	400	138	235	469	556	116×113	106	196
600	600	057	155	549	636	173×169	006	096

Tab. 3.15 Größe und Verschiebung des Bildes *bild2.ps* ($1pt = 0.35146mm$)

3.8.2 Kreis und rechtwinkeliges Dreieck

Die Fläche des Einheitkreises $F_\circ = \pi r^2,\ r = 1$, lässt sich transformieren auf die Fläche eines rechtwinkeligen Dreiecks $F_\triangle = \frac{1}{2} r \cdot 2\pi r,\ r = 1$.
Die Darstellungen des Sachverhalts beruht auf flächendeckenden Kurven (Mäander). Man stelle sich den Einheitskreis vor überdeckt von vielen konzentrischen Kreisen in Form von "dicken Wollfäden". Diese sind an ihrem Anfang auf der Strecke $x = [0, 1]$ festgemacht und das andere Ende ist frei. Durch eine vertikale Luftströmung von unten kommend richten sich die Fäden auf und bilden flächendeckend ein rechtwinkeliges Dreieck.

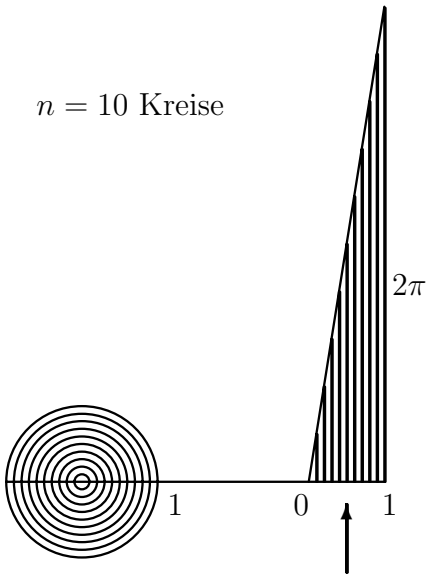


Abb. 3.20 Einheitkreis = rechtwinkeliges Dreieck

Nun wollen wir die Vorgehensweise zur Darstellung der Flächenübereinstimmung in Maple implementieren, wobei wir mit einfachen Figuren beginnen und bei der Animation für das "Aufrichten" der Fäden enden werden.

Rechnungen in Maple

Formel der Kreisflaeche $F_\circ = \pi r^2$ aus Umfang und rechtwinkeligem Dreieck

```
> restart:
  with(plots):
  with(plottools):
  with(plots,animate):
```

Kreisgleichung, Einheitskreis

```
> r:=1:
  u:=[r*cos(x),r*sin(x),x=0..2*Pi]:
  plot(u,thickness=3,color=black,scaling=constrained);
```

$n = 10$ Kreise (flaechendeckend, Maeanderisierung)

```
> n:=10:
  u:=r->[r*cos(x),r*sin(x),x=0..2*Pi]:
  plot([seq(u(i/n),i=1..n)],thickness=15,scaling=constrained,
        color=[seq(COLOR(RGB,2*i/n*rand()/10^12,
                             2*i/n*rand()/10^12,
                             2*i/n*rand()/10^12),i=1..n)]);

> pl1:=plot([seq(u(i/n),i=1..n)],thickness=15,scaling=constrained,
              color=[seq(COLOR(RGB,2*i/n*rand()/10^12,
                                   2*i/n*rand()/10^12,
                                   2*i/n*rand()/10^12),i=1..n)]):

> datei1:=`C:/D/neundorf/maple3/kkcol1.ps`:
  interface(plotdevice=ps,plotoutput=datei1,
            plotoptions=`color,portrait,noborder`);
  plots[display](pl1);
  interface(plotdevice=win);
```

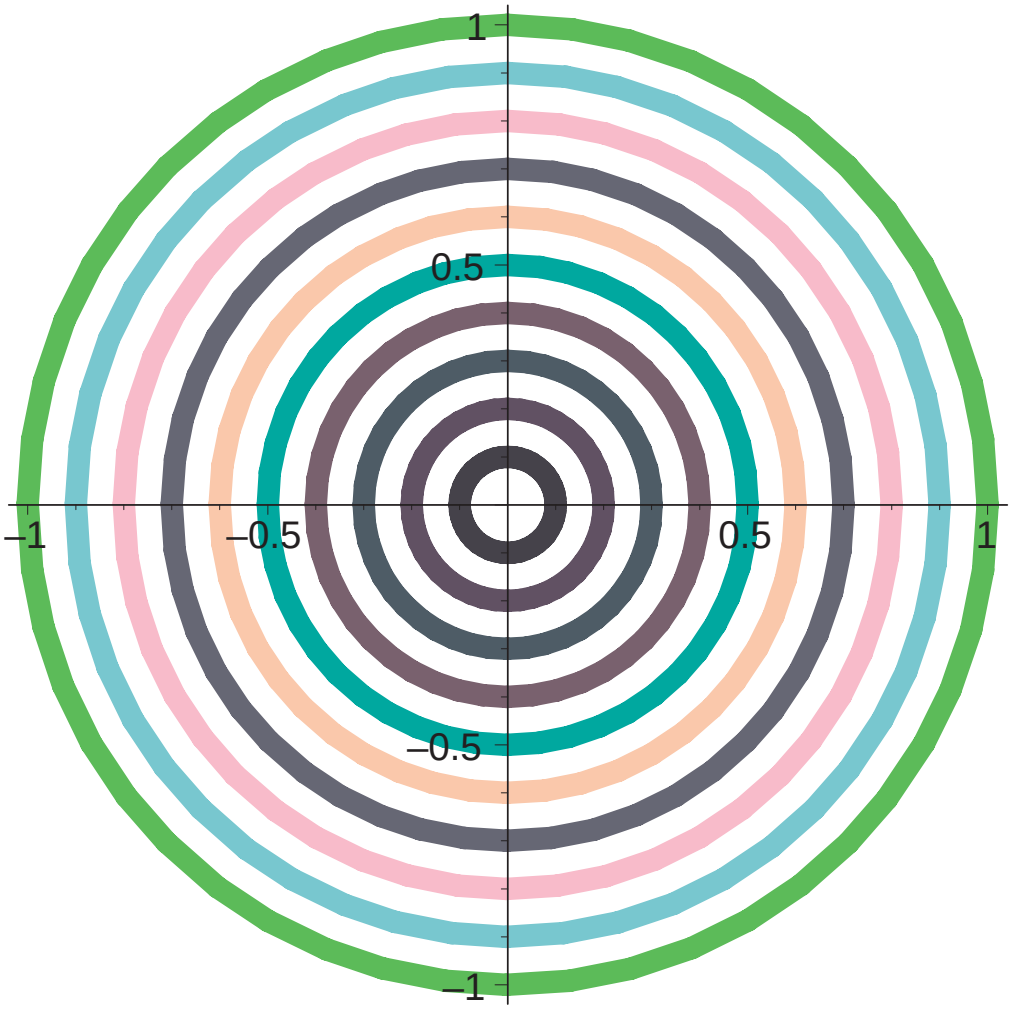


Abb. 3.21 $n = 10$ Kreise (flächendeckend, Mäanderisierung)

Animation: Aufrichten des Einheitskreises zu einer vertikalen Strecke

```
> animate([(1-t)*cos(x)+t,(1-t)*sin(x)+x*t,x=0..2*Pi],t=0..1,
          scaling=constrained,frames=16,color=gold,thickness=12);

> plots[display](op(1, op(1,%)),scaling=constrained,
       thickness=12,view=[-1..1,-1..6.3]);    # 1. Frame
  plots[display](op(16,op(1,%%)),scaling=constrained,
       thickness=12,view=[-1..1,-1..6.3]);    # letztes Frame
```

Einheitskreis ($t = 0$) bis vertikale Strecke der Laenge 2π bei $0 \leq x \leq 2\pi$

```
> v:=(t,x)->(1-t)*cos(x)+t:
  w:=(t,x)->(1-t)*sin(x)+x*t:
```

Bildfolge von Kreis ($t = 0$) mit Radius r bis vertikale Strecke der Laenge $2\pi r$ fuer alle n Kreise

```
> m:=15:   # m+1 = Anzahl der Frames
  p1:=(k,j)->plot([k/n*v(j/m,x),k/n*w(j/m,x),x=0..2*Pi], # r=k/n, k=1..n
            scaling=constrained,thickness=4,
            color=COLOR(RGB,(m-j+1)/(m+3),(j+1)/(m+3),(m-j+1)/(m+3))):
```

Aeusserer Kreis (Nummer n)

```
> p1(n,1);     # plots[display](p1(n,1));   # 2. Frame nach Kreis

> p2:=[seq(p1(n,j),j=0..m)]:
  # Animation
  plots[display](p2,insequence=true,view=[-1..1,-1..6.3],
                   scaling=constrained);
```

Bildfolge fuer den aeusseren Kreis (16 Frames) mit Darstellung
1. Version

```
> m2:=(m+1)/2:
  ph:=array(1..m+1,[]):
  pp:=array(1..2,1..m2,[]):
  for l from 0 to m do
   ph[l+1]:=display(p1(n,l)):
  end do:
  for l from 1 to m2 do
   pp[1,l]:=display(ph[l],tickmarks=[0,0]):
   pp[2,l]:=display(ph[m2+l],tickmarks=[0,0]):
  end do:

  plots[display](ph);   # 16 Bilder nebeneinander, zu eng
  plots[display](pp);   # 16 Bilder als Tableau 2*8, guenstiger

> datei2:=‘C:/D/neundorf/maple3/kkcol2.ps‘:
  interface(plotdevice=ps,plotoutput=datei2,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](ph);
  interface(plotdevice=win);
```

Abb. 3.22 Bildfolge für den äußeren Kreis ($1 * 16$ Frames), ungünstige Darstellung, durch Skalierung des *ps*-Files nicht besser

```
> datei3:=‘C:/D/neundorf/maple3/kkcol3.ps‘:
  interface(plotdevice=ps,plotoutput=datei3,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp);
  interface(plotdevice=win);
```

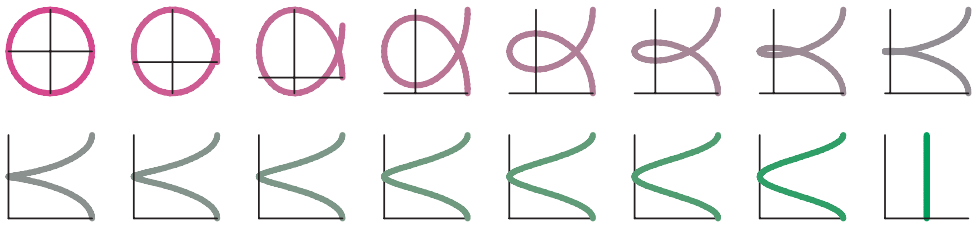


Abb. 3.23 Bildfolge für den äußeren Kreis ($2 * 8$ Frames)

2. Version

```
> p1m:=(k,j)->plot([k/n*v(j/m,x),k/n*w(j/m,x),x=0..2*Pi], # [1,6.3] },
              scaling=constrained,thickness=4,tickmarks=[0,0],
              view=[-3.5..3.5,-1..6.3],
              color=COLOR(RGB,(m-j+1)/(m+3),(j+1)/(m+3),(m-j+1)/(m+3))):
  pz:=plot([[3.5,-1],[3.5,6.3],[-3.5,6.3]],color=white):
              # Zwangsmassnahme
  p1mm:=(k,j)->display([p1m(k,j),pz],scaling=constrained):
  display(p1mm(n,0));
  display(p1mm(n,m));

> m4:=(m+1)/4:
  pp:=array(1..4,1..m4,[]):
  for l from 1 to m4 do
   pp[1,l]:=display(p1m(n,l-1)):
   pp[2,l]:=display(p1m(n,m4+l-1)):
   pp[3,l]:=display(p1m(n,2*m4+l-1)):
   pp[4,l]:=display(p1m(n,3*m4+l-1)):
  end do:
  plots[display](pp);   # 4*4 Bilder nebeneinander, nicht skaliert

> datei4:=‘C:/D/neundorf/maple3/kkcol4.ps‘:
  interface(plotdevice=ps,plotoutput=datei4,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp);
  interface(plotdevice=win);
```

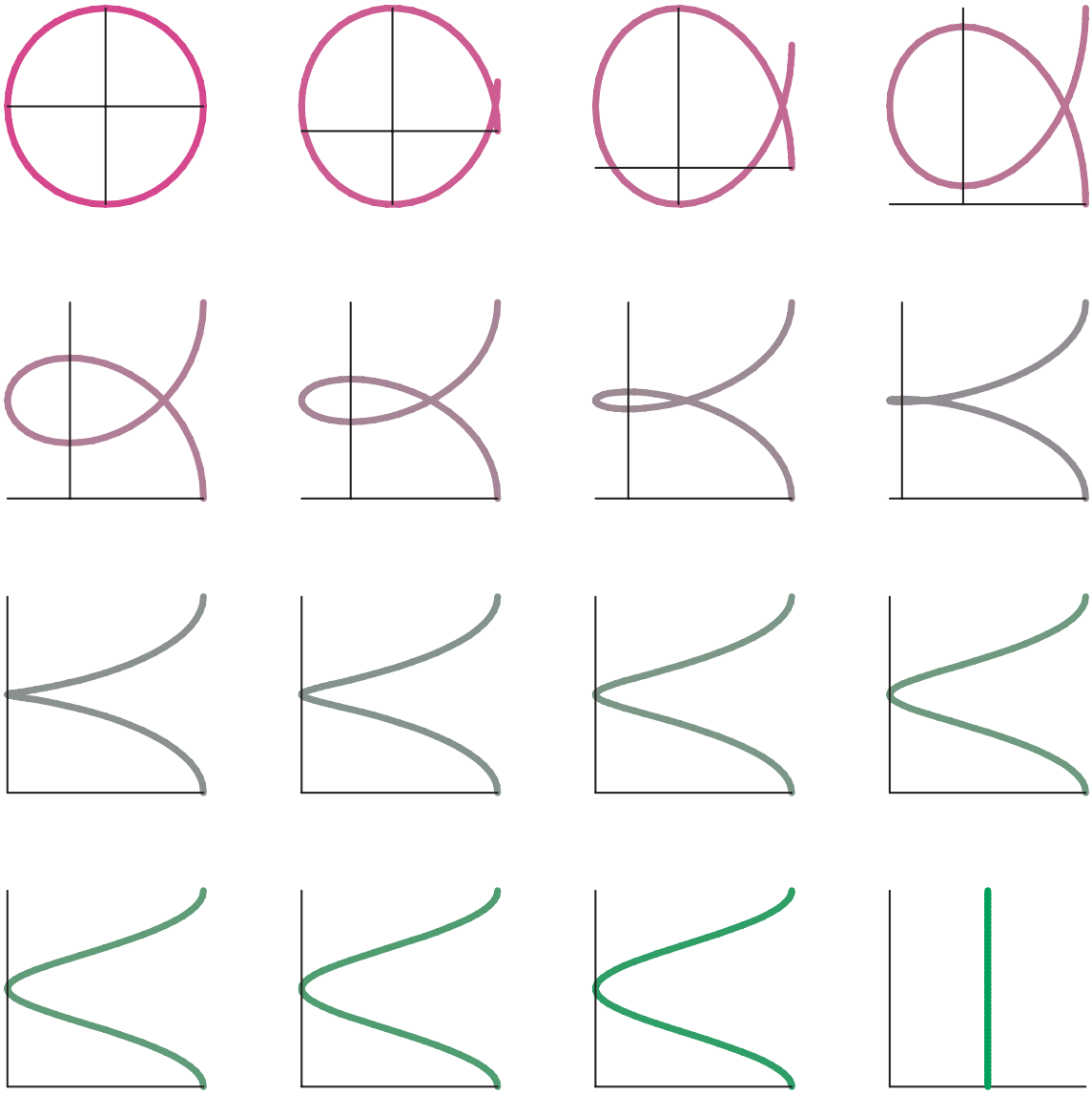


Abb. 3.24 Bildfolge für den äußeren Kreis ($4 * 4$ Frames, nicht skaliert)

```
> for l from 1 to m4 do
   pp[1,l]:=display(p1mm(n,l-1)):
   pp[2,l]:=display(p1mm(n,m4+l-1)):
   pp[3,l]:=display(p1mm(n,2*m4+l-1)):
   pp[4,l]:=display(p1mm(n,3*m4+l-1)):
  end do:
  plots[display](pp);   # 4*4 Bilder nebeneinander, skaliert

> datei5:=‘C:/D/neundorf/maple3/kkcol5.ps‘:
  interface(plotdevice=ps,plotoutput=datei5,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp);
  interface(plotdevice=win);
```

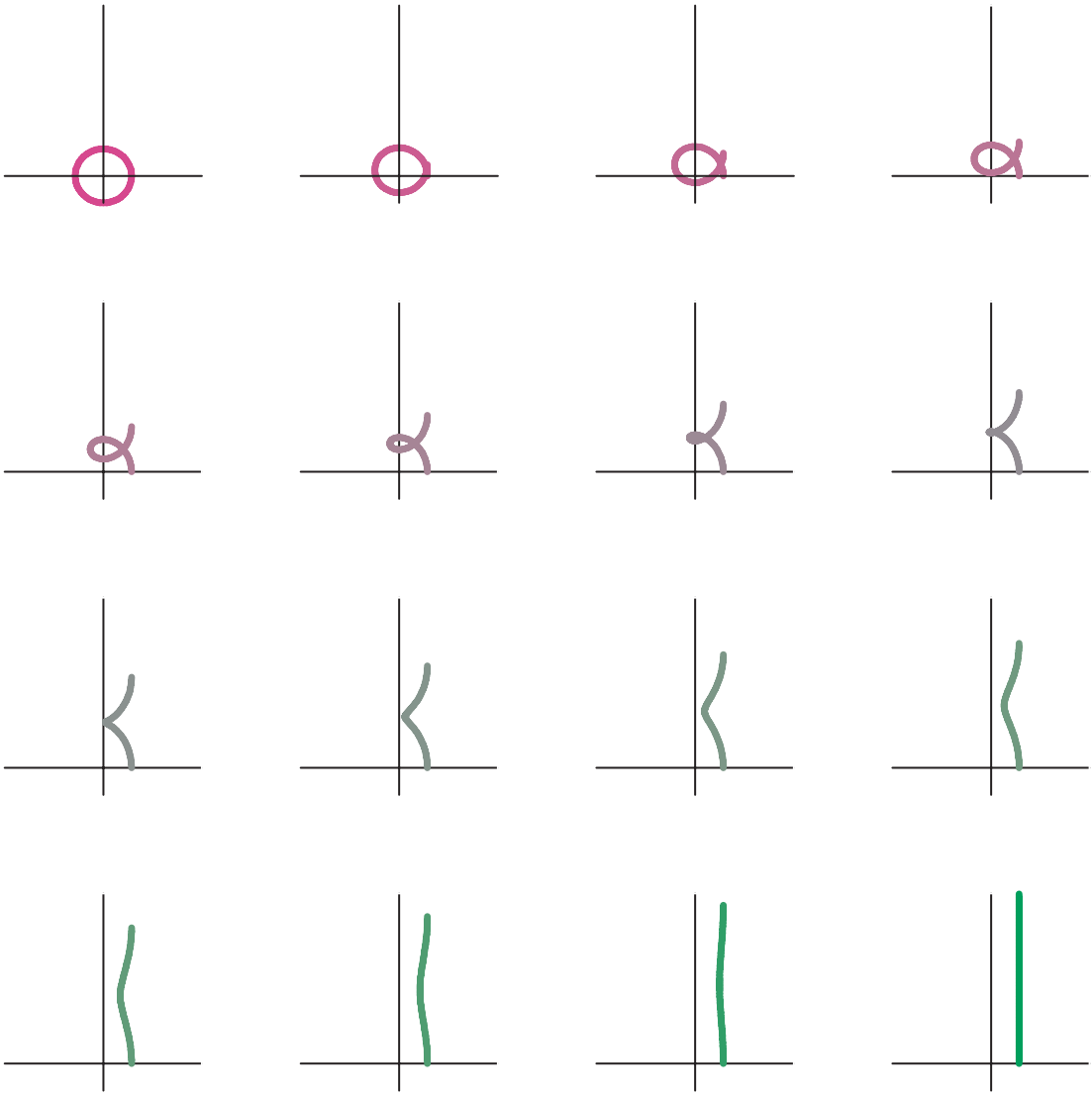


Abb. 3.25 Bildfolge für den äußeren Kreis ($4 * 4$ Frames, skaliert)

1. Frame fuer alle n Kreise

```
> p1(1,1);        # plots[display](p1(1,1));
```

```
> p3:=[seq(p1(k,1),k=1..n)]:
  # Animation
  plots[display](p3,insequence=true,thickness=4,scaling=constrained);
```

Bildserie des 1. Frames fuer alle 10 Kreise

```
> pz:=plot([[1,1],[-1,1],[-1,-1]],color=white):
  ph:=array(1..n,[]):
  for l from 1 to n do
   ph[l]:=display([p1(l,1),pz],view=[-1..1,-1..1],tickmarks=[0,0]):
  end do:
  plots[display](ph);   # n=10 Bilder nebeneinander
```

```
> datei6:=‘C:/D/neundorf/maple3/kkcol6.ps‘:
  interface(plotdevice=ps,plotoutput=datei6,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](ph);
  interface(plotdevice=win);
```

Abb. 3.26 1. Frame für alle 10 Kreise

Bild mit allen Kreisen und Frames

```
> ph:=array(1..m+1,[]):
  pp:=array(1..4,1..m4,[]):

  for l from 0 to m do
   ph[l+1]:=display(seq(p1(k,l),k=1..n)):
  end do:

  for l from 1 to m4 do
   pp[1,l]:=display(ph[l],tickmarks=[0,0]):
   pp[2,l]:=display(ph[m4+l],tickmarks=[0,0]):
   pp[3,l]:=display(ph[2*m4+l],tickmarks=[0,0]):
   pp[4,l]:=display(ph[3*m4+l],tickmarks=[0,0]):
  end do:

> p4:=[seq(seq(p1(k,j),k=1..n),j=0..m)]:

  # Animation
  # 1.-16. Frame ueber jeweils alle 10 Kreise
  plots[display](p4,view=[-1..1,-1..6.3],
        insequence=true,
        scaling=constrained);

> plots[display](ph);   # 16 Bilder nebeneinander

> plots[display](pp);   # 16 Bilder als Tableau 4*4

> datei7:=‘C:/D/neundorf/maple3/kkcol7.ps‘:
  interface(plotdevice=ps,plotoutput=datei7,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp);
  interface(plotdevice=win);
```

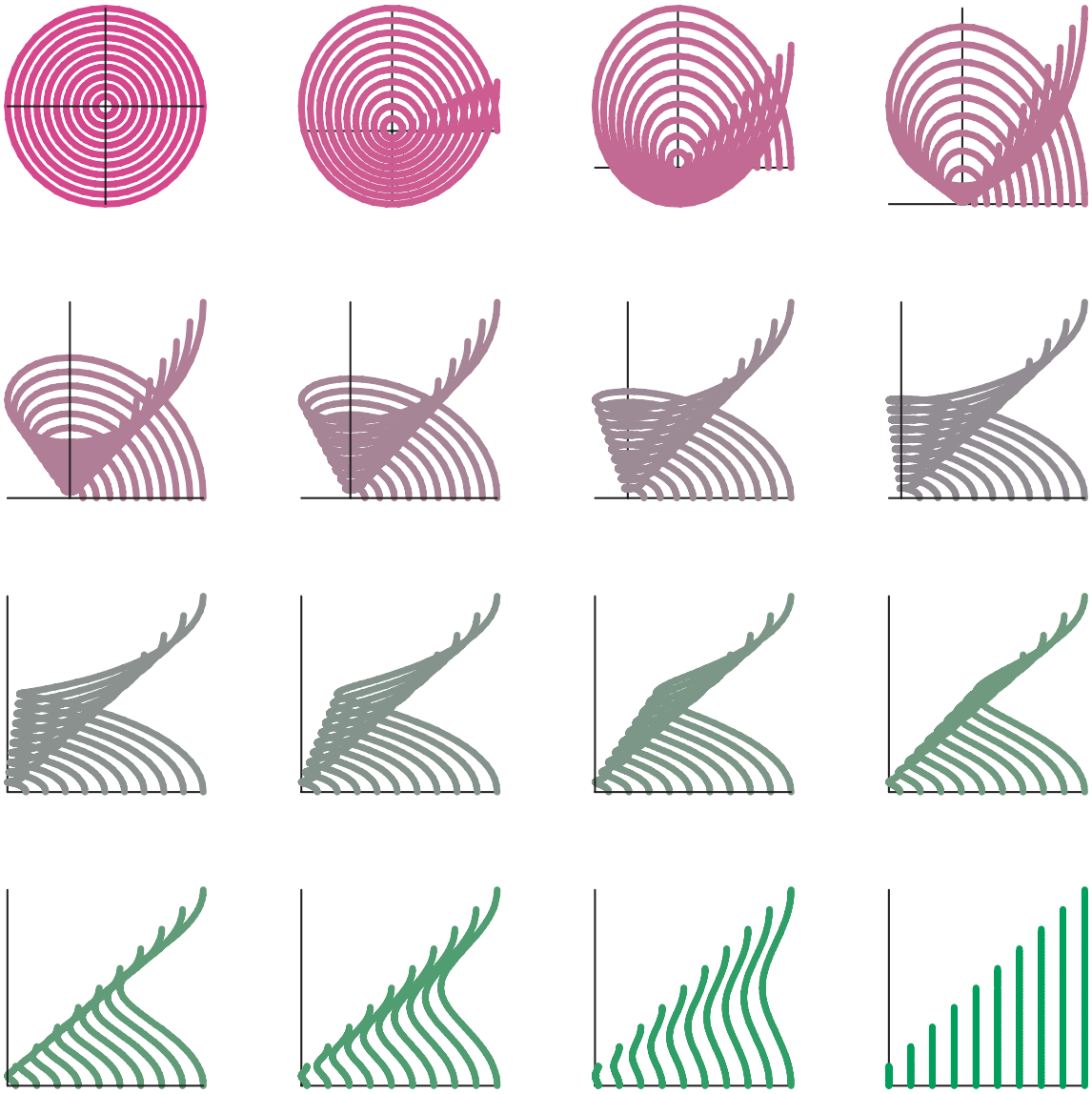


Abb. 3.27 Alle 16 Frames für alle 10 Kreise

Animation der “Aufspreizung“, Faeden etwas enger

```
> p11:=(k,j)->plot([k/n*v(j/m,x),k/n*w(j/m,x),x=0..2*Pi],
              scaling=constrained,thickness=6,
              color=COLOR(RGB,(m-j+1)/(m+3),(j+1)/(m+3),(m-j+1)/(m+3))):
  ph1:=array(1..m+1,[]):
  for l from 0 to m do
   ph1[l+1]:=display(seq(p11(k,l),k=1..n)):
  end do:

> picts:=[seq(ph1[l],l=1..m+1)]:
  display(picts,insequence=true,scaling=constrained,
          view=[-1..1,-1..6.3],title=‘F=r*(2*r*Pi)/2=Pi*r^2‘);
```

1. Frame (alle Kreise) und 16. Frame (alle Vertikalen, rechtwinkeliges Dreieck)

```
> display(ph1[1],ph1[m+1]);
> datei8:=‘C:/D/neundorf/maple3/kkcol8.ps‘:
  interface(plotdevice=ps,plotoutput=datei8,
  plotoptions=‘color,portrait,noborder‘);
  plots[display](ph1[1],ph1[m+1]);
  interface(plotdevice=win);
```

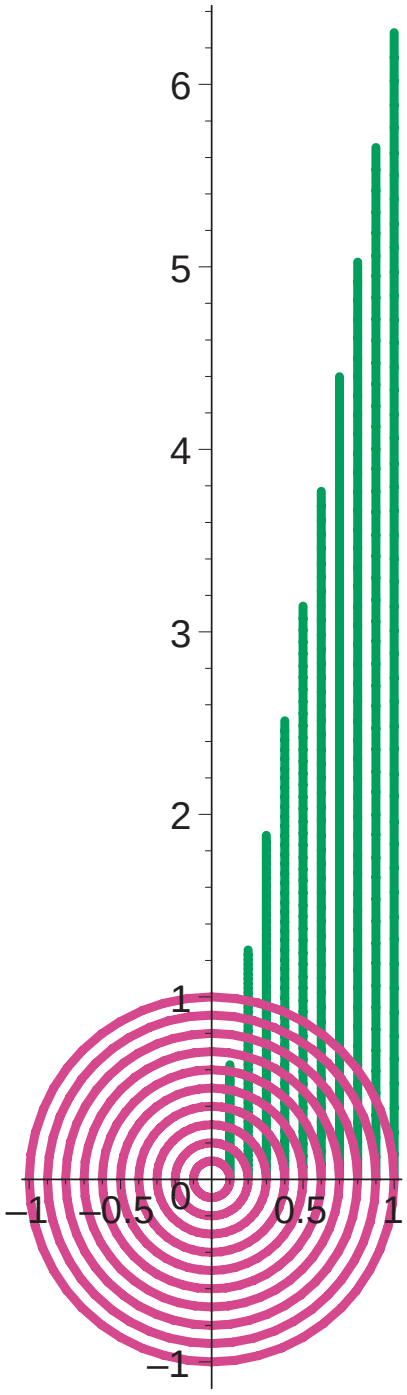


Abb. 3.28
1. Frame (alle Kreise) und 16. Frame (alle Vertikalen)

3.8.3 Quadratwurzel mittels Höhensatz

Der Höhensatz von Euklid im rechtwinkeligem Dreieck sagt, dass das Quadrat über der Höhe gleich dem Produkt der beiden anliegenden Hypotenuseabschnitten ist: $h^2 = pq$.

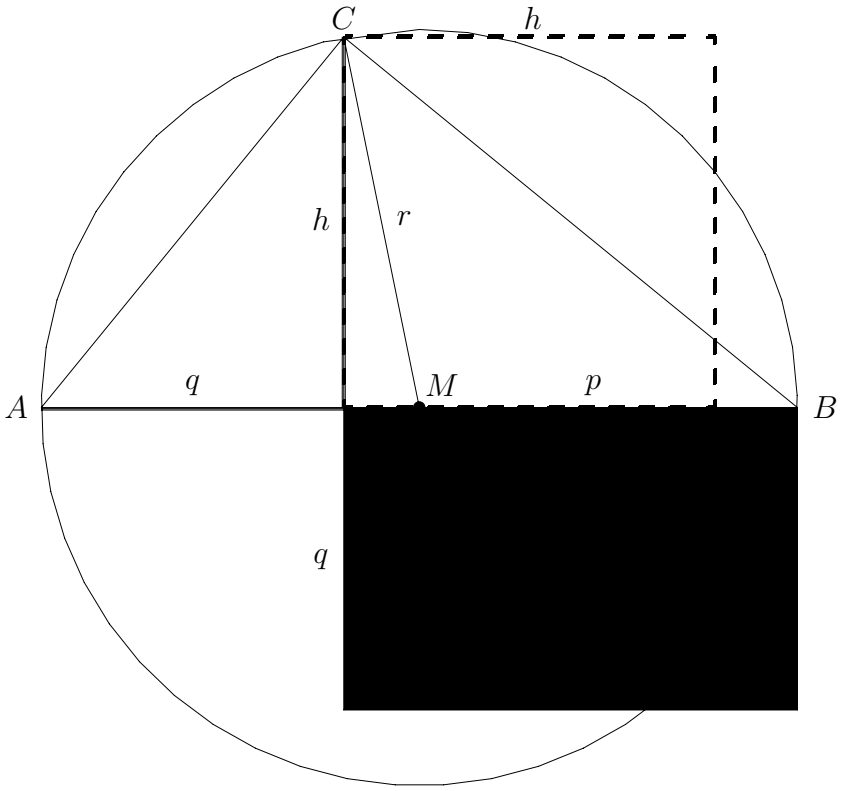


Abb. 3.29 Datei *hoehe*1.*pic*, Höhensatz von Euklid

Ähnlich kann man den Kathetensatz von Euklid oder den Satz von Pythagoras darstellen.
Wir wollen aber den Höhensatz zur Quadratwurzelberechnung aus einer gegebenen Zahl $t > 0$ verwenden. Dazu setzen wir im Höhensatz $t = p$ und $q = 1$ und erhalten somit $h = \sqrt{t}$, was sich nun in Maple schön demonstrieren lässt.

Rechnungen in Maple

```
> restart:
  with(plots):

> # Radikand
  t:=5;
  'sqrt(t)'=sqrt(t);
  'sqrt(t)'=sqrt(5.0);
```

```
> # Teilplots
  pl1:=plot([[0,0.03],[t,0.03]],-1..t,thickness=4,color=blue):
  pl2:=plot([[-1,0.03],[0,0.03]],-1..t,thickness=4,color=green):
  x0:=(t-1)/2:
  r:=(t+1)/2:
  h:=sqrt(r^2-(0-x0)^2):
  pl3:=plot(sqrt(r^2-(x-x0)^2),x=-1..t,thickness=2,color=red):
  pl4:=plot([[-1,0],[0,h],[t,0]],thickness=1,color=red):
  pl5:=plot([[0.02,0],[0.02,h]],thickness=4,color=red):
  pl6:=plot([[x0,-0.05],[x0,0.2]],thickness=2,color=blue):
  pl7:=plot([[x0,0]],style=point,symbol=circle,color=blue):

  st:=convert(t,string):
  hk:=sqrt(r^2-(x-x0)^2):
  shk:=convert(hk,string):
  shk:=cat(‘f(x)=sqrt(9-(x-2)^2)=‘,shk):
  h:=’h’:
  p1:=textplot([[1,1,‘h=f(0)=sqrt(5)‘],[2.5,0.3,‘t=‘||st],[3,3.2,shk]],
               font=[TIMES,ITALIC,10]):
  pp:=display([pl3,pl4,pl6,pl7,p1,pl5,pl1,pl2],scaling=constrained,
  title=‘Hoehensatz im rechtwinkeligen Dreieck h^2=1*t --> Wurzel aus t‘):
  display(pp);
```

$$t := 5$$
$$\sqrt{t} = \sqrt{5}$$
$$\sqrt{t} = 2.236067977$$

```
> datei1:=‘C:/D/neundorf/maple3/geom1.ps‘:
  interface(plotdevice=ps,plotoutput=datei1,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](pp);
  interface(plotdevice=win);
```

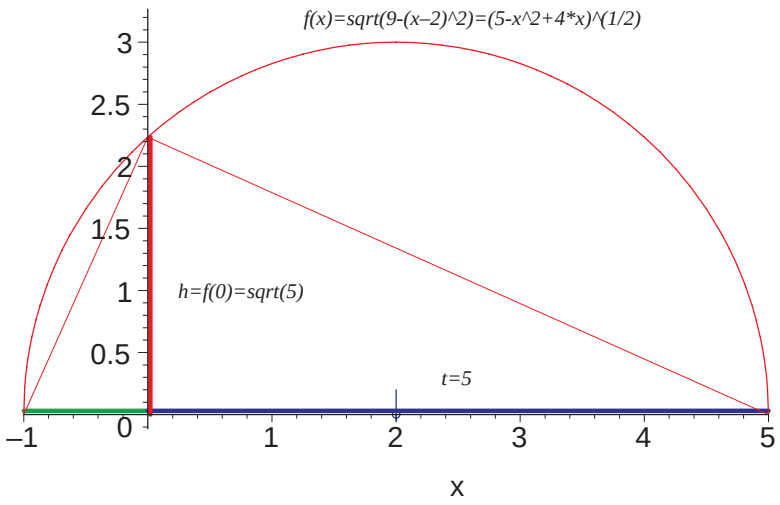


Abb. 3.30 Quadratwurzel $h = \sqrt{t},\ t > 0$, mittels Höhensatz

3.8.4 Uhrzeiger mit einer Umdrehung

Das Ziffernblatt einer Uhr mit Sekundenzeiger sei die Ausgangssituation für die folgende einfache Betrachtung. Wir wollen die 60 Positionen des Sekundenzeigers als Animation in Maple darstellen.

Rechnungen in Maple

```
> restart:
  with(plots):
```

Uhr mit Sekundenzeigerstellungen

```
> kreis:=plot([sin(t),cos(t),t=0..2*Pi],-1..1,-1..1,
              scaling=constrained,axes=none):
  zeiger:=t->plot([[0,0],[sin(t),cos(t)]],thickness=3,tickmarks=[0,0]):

  n:=60:
  drehz:=[seq(zeiger(2*Pi*i/n),i=0..n)]:
  # Animation
  pdreh:=plots[display](drehz,insequence=true,scaling=constrained):
  plots[display](kreis,pdreh);
```

Alle Zeigerstellungen

```
> plots[display](kreis,drehz);
```

```
> datei2:=‘C:/D/neundorf/maple3/geom2.ps‘:
  interface(plotdevice=ps,plotoutput=datei2,
            plotoptions=‘color,portrait,noborder‘);
  plots[display](kreis,drehz);
  interface(plotdevice=win);
```

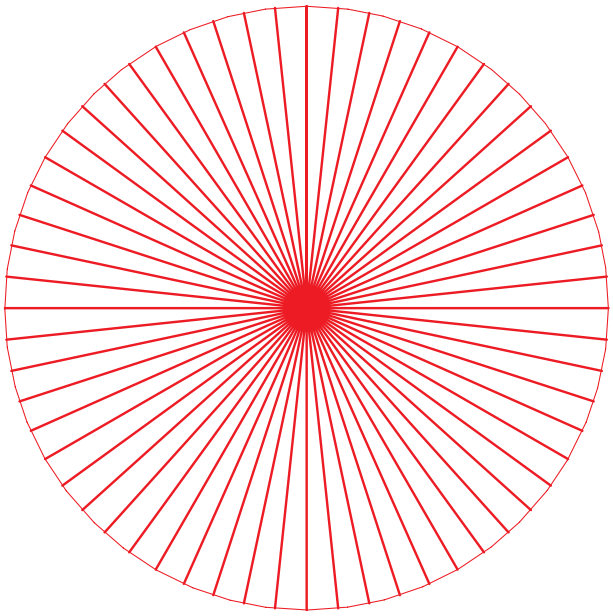


Abb. 3.31
Ziffernblatt einer Uhr mit 60 Sekundenzeigerstellungen

3.8.5 Tannenbäume mit Weihnachtskugel bzw. Osterei

Um eine Fichte oder Tanne relativ abstrakt darzustellen, benötigt man die Sägezahnkurve, die mit vertikaler und/oder horizontaler Stauchung/Streckung bearbeitet sowie ihren gespiegelten Teilen verbunden wird. Dazu bieten sich die mathematischen Standardfunktionen `floor, ceil, frac, trunc, round` an. Wir verwenden die Funktion, die von einer Zahl x die größte ganze Zahl $z \leq x$ bestimmt, also `floor(x)`. Sie ist die sogenannte Treppenfunktion, mit deren Hilfe gemäß $x - \text{floor}(x) \in [0, 1]$ bzw. $x - \text{floor}(x) - 1 \in [-1, 0]$ die gewünschten "Sägezähne" entstehen. Nimmt man an Stelle von x eine andere Funktion, so kann damit schon eine horizontale Stauchung erfolgen. Vertikale Veränderungen erreicht man durch Dämpfungsfunktionen wie z. B. e^{-x} oder $a - x$, falls $0 \leq x \leq a$ ist.
Versuchen wir nun, einige geeignete "Baumfunktionen" zu finden.

Rechnungen in Maple

4 Baumfunktionen

1. Funktion

```
> f1:=x->tan(x)-1-floor(tan(x)); # eine Baumseite
  f2:=x->-f1(x);                 # und Spiegelung
```

$$f1 := x \rightarrow \tan(x) - 1 - \text{floor}(\tan(x))$$
$$f2 := x \rightarrow -\text{f1}(x)$$

```
> p1:=plot([tan(x)-1,f1(x),f2(x)],x=0..Pi/2,y=-1..4,thickness=[1,2,2],
           color=[black,green,green]):
  display(p1);
```

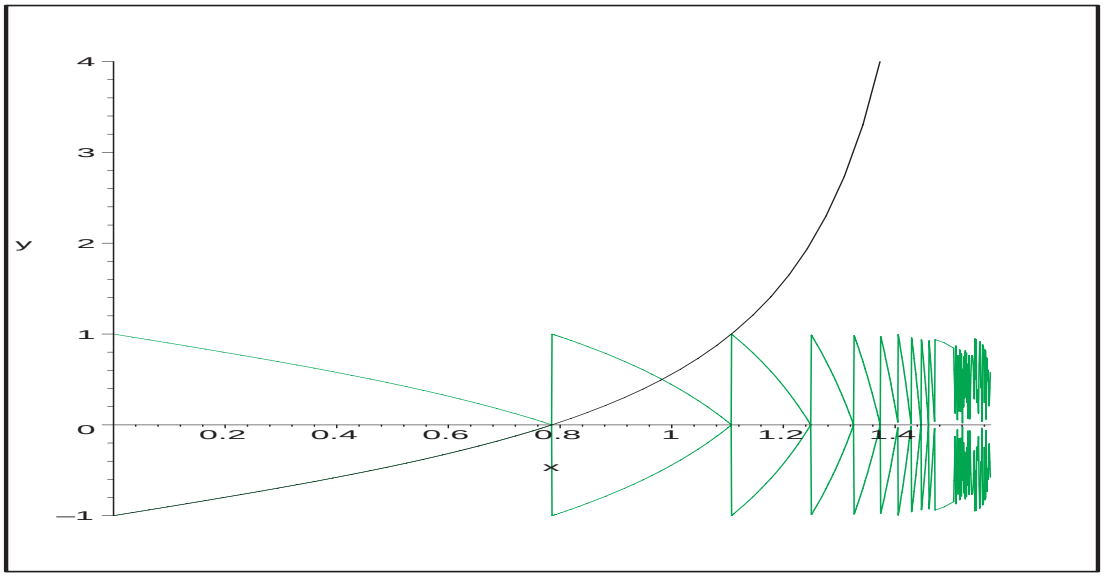


Abb. 3.32 Grundlagen für Baumfunktion

```
> datei3:='C:/D/neundorf/maple3/geom3.ps':
  interface(plotdevice=ps,plotoutput=datei3,
            plotoptions='color,portrait');
  plots[display](p1);
  interface(plotdevice=win);
```

Verjuengung des Baums zur Spitze hin

```
> f3:=x->(Pi/2-x)*(0.9*f1(x)-0.1);
  f4:=x->(Pi/2-x)*(0.9*f2(x)+0.1);
```

$$f3 := x \to \left(\frac{\pi}{2} - x\right)(0.9\,\mathrm{f1}(x) - 0.1)$$
$$f4 := x \to \left(\frac{\pi}{2} - x\right)(0.9\,\mathrm{f2}(x) + 0.1)$$

```
> p2:=plot([f3(x),f4(x)],x=0..Pi/2,y=-Pi/2..Pi/2,thickness=[2,2],
            color=[green,green]):
  plots[display](p2);
```

```
> datei4:='C:/D/neundorf/maple3/geom4.ps':
  interface(plotdevice=ps,plotoutput=datei4,
            plotoptions='color,portrait');
  plots[display](p2);
  interface(plotdevice=win);
```

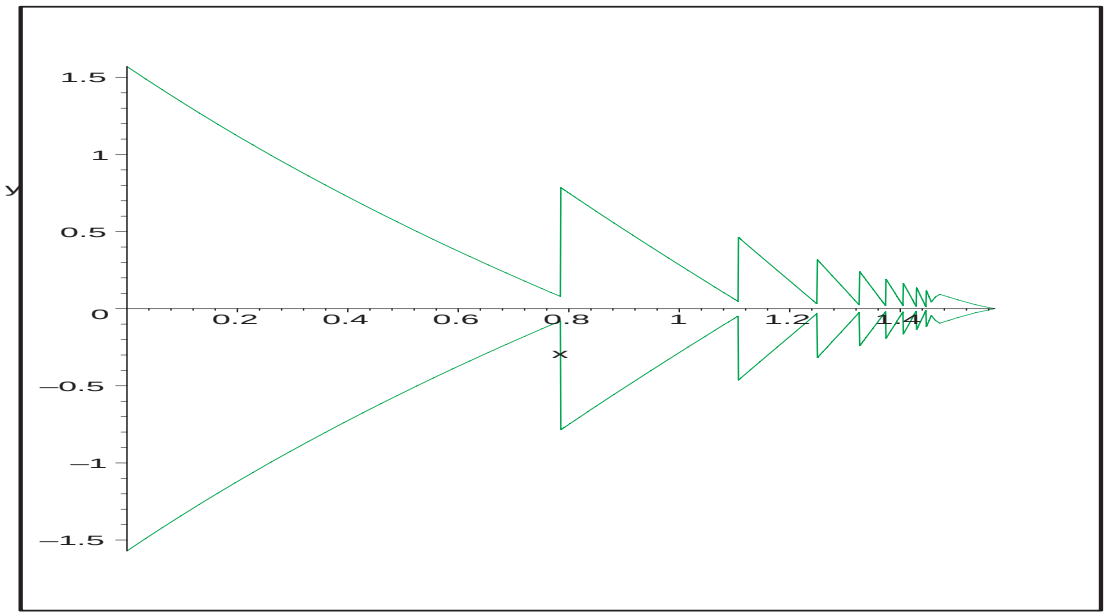


Abb. 3.33 Verjüngung des Baums zur Spitze hin

Aufrichten des Baums

```
> p3:=plot([f3(x),x,x=0..Pi/2],thickness=3,color=green):
  display(p3,axes=none);

> datei5:=‘C:/D/neundorf/maple3/geom5.ps‘:
  interface(plotdevice=ps,plotoutput=datei5,
            plotoptions=‘color,portrait‘);
  plots[display](p3,axes=none);
  interface(plotdevice=win);
```

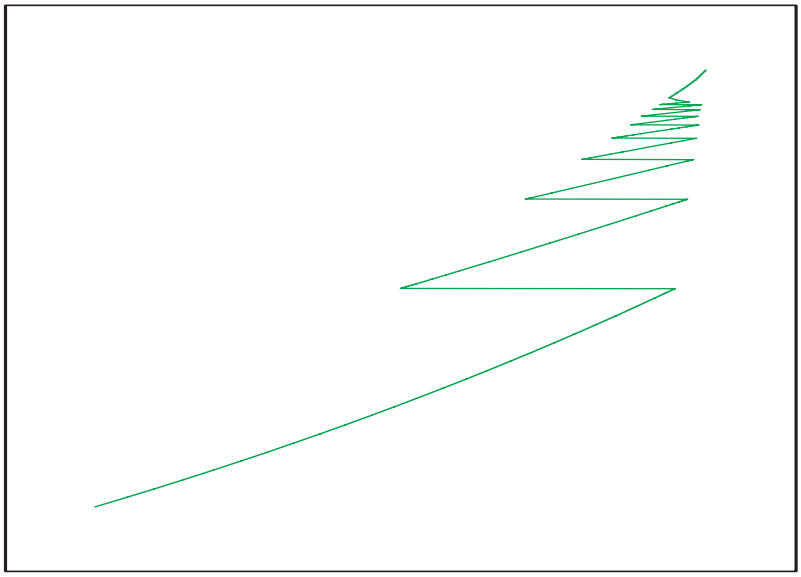


Abb. 3.34 Aufrichten des Baums

Gesamtgestaltung des Baums mit Stamm und Schmuck

```
> q1:=plot([f3(x),x,x=0..Pi/2],thickness=3,color=green):
  q2:=plot([f4(x),x,x=0..Pi/2],thickness=3,color=green):
  q3:=plot(0,x=-Pi/2..Pi/2,thickness=3,color=green):
  q4:=plot(0.62+sqrt(0.01-(x-0.75)^2),x=0.65..0.85,thickness=3,color=red):
  q5:=plot(0.62-sqrt(0.01-(x-0.75)^2),x=0.65..0.85,thickness=3,color=red):
  q6:=plot([[-0.1,0],[-0.1,-0.3],[0.1,-0.3],[0.1,0]],
            thickness=4,color=brown):
  q7:=plot([[0.75,0.73],[0.75,0.81]],thickness=3, color=red):

> p4:=plots[display](q1,q2,q3,q4,q5,q6,q7,axes=none,
            title=‘Frohe Weihnachten/Ostern‘):
  display(p4);

> datei6:=‘C:/D/neundorf/maple3/geom6.ps‘:
  interface(plotdevice=ps,plotoutput=datei6,
            plotoptions=‘color,portrait‘);
  plots[display](p4);
  interface(plotdevice=win);
```

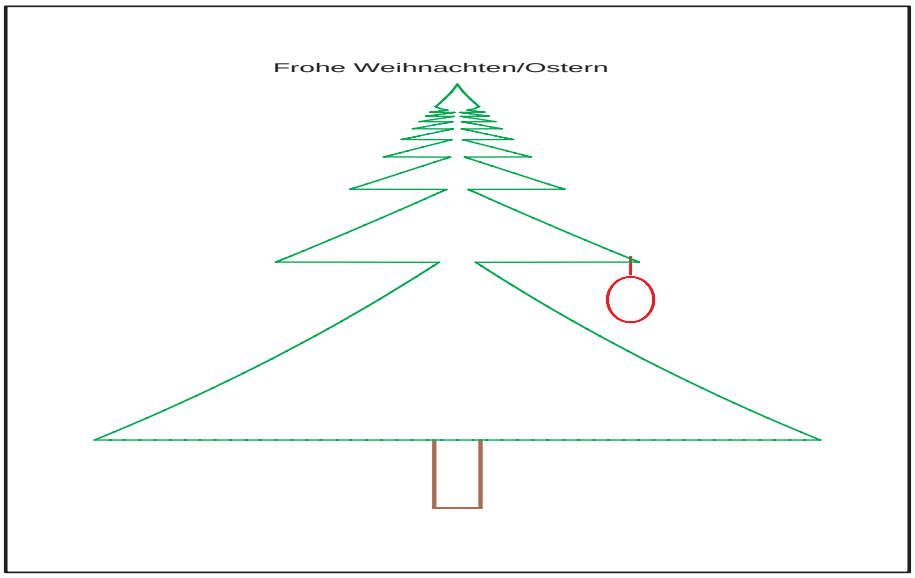


Abb. 3.35 Gesamtgestaltung des Baums mit Stamm und Schmuck

Man probiere es selbst

2. Funktion

```
> g1:=x->cot(x)-floor(cot(x));
  g2:=x->-g1(x);
```

$$g1 := x \to \cot(x) - \text{floor}(\cot(x))$$
$$g2 := x \to -\text{g1}(x)$$

```
> plot([cot(x),g1(x),g2(x)],x=0..Pi/2-1E-4,y=-1..4,thickness=[1,2,2],
        color=[black,green,green]);
```

3. Funktion

```
> h1:=x->1/x-floor(1/x);
  h2:=x->-h1(x);
```

$$h1 := x \to \frac{1}{x} - \text{floor}\left(\frac{1}{x}\right)$$
$$h2 := x \to -\text{h1}(x)$$

```
> plot([1/x,h1(x),h2(x)],x=0..1,y=-1..4,thickness=[1,2,2],
        color=[black,green,green]);
```

4. Funktion

```
> k1:=x->exp(x)-1-floor(exp(x));
  k2:=x->-k1(x);
```

$$k1 := x \to e^x - 1 - \text{floor}(e^x)$$
$$k2 := x \to -\text{k1}(x)$$

```
> plot([exp(x)-1,k1(x),k2(x)],x=0..2.078,y=-1..4,thickness=[1,2,2],
        color=[black,green,green]);
```

Kapitel 4

Was leistet der Computer?

An solchen Beispielen zeigt sich, was ein Computer schon und noch nicht leisten kann. Der wirklich kreative Anteil am Problemlösungsprozess bleibt beim Menschen. Der Phantasie sind aber keine Grenzen gesetzt. Natürlich ist es zumeist nicht einfach, komplizierte Sachverhalte methodisch geschickt und didaktisch wirksam für Präsentationen vor einem Hörerkreis aufzubereiten.
Denkt man bei Computerunteranwendungen zum Beispiel an adaptive und selbstkorrigierende Verfahren, Intervallarithmetik, Konzept der hohen und optimalen Genauigkeit, wissenschaftliches Rechnen mit Ergebnisverifikation oder an paralleles und verteiltes Rechnen (parallel and distibuted computing), so sind hier schon neue und erfolgversprechende Wege gegangen worden.
So erhält man mit Systemen des wissenschaftlichen Rechnens, wie es zum Beispiel Pascal-XSC ist, neben einem Ergebnis auch sehr nützliche Informationen über seine Bewertung und Brauchbarkeit.
Solche und noch nicht vorstellbare Entwicklungen sind ernst zu nehmen, um die komplexen und wachsenden Probleme zu meistern.

Literaturverzeichnis

Maple

[1] Blachman, N. und Mossinghoff M. J.: *Maple griffbereit*. Vieweg Verlag Braunschweig 1996.

[2] Heal, K. M. et al.: *Handbuch Waterloo Maple*. Maple V - Learning Guide, Springer-Verlag New York 1998.

[3] Heck, A.: *Introduction to Maple*. 2nd Ed. Springer-Verlag New York 1996.

[4] Krawietz, A.: *Maple V für das Ingenieurstudium*. Springer-Verlag New York 1997.

[5] Klimek, G. und M. Klimek: *Discovering Curves and Surfaces with Maple*. Springer-Verlag New York 1997.

[6] Kofler, M.: *Maple V Release 4*. Addison Wesley Bonn 1996.

[7] Monagan, M.: *Programming in Maple: The Basics*. Institut für Wissenschaftliches Rechnen ETH-Zentrum, CH-8092 Zürich.

[8] Monagan, M. et al.: *Maple V Programming Guide*. Springer-Verlag New York 1996.

[9] Neundorf, W.: *Programming in Maple V Release 5*. Extended Basics. Preprint M 07/99 IfMath der TU Ilmenau, Februar 1999.

[10] Neundorf, W. und B. Walther: *Grafik, Animation und Ausgabeformate in Maple V Release 5*. Preprint M 12/00 IfMath der TU Ilmenau, Juni 2000.

[11] Nicolaides, R. A. und N. J. Walkington: *Maple: A Comprehensive Introduction*. Cambridge University Press 1996.

[12] Walz, A.: *Maple V. Rechnen und Programmieren mit Release 4*. R. Oldenbourg Verlag München 1998.

[13] Werner, W.: *Mathematik lernen mit Maple*. Bd. 1,2. Ein Lehr-und Arbeitsbuch für das Grundstudium. dpunkt Heidelberg 1996, 1998 (CD).

[14] Westermann, T.: *Mathematik für Ingenieure mit Maple*. Springer-Verlag 1996.

Matlab

[15] Chen, K., P. J. Giblin und A. Irving: *Mathematical explorations with MATLAB*. Cambridge University Press 1999.

[16] Golubitsky, M. und M. Dellnitz: *Linear algebra and differential equations using MATLAB*. Brooks/Cole Pub. Co Pacific Grove 1999.

[17] Gramlich, G. und W. Werner: *Numerische Mathematik mit Matlab*. 1. Auflage. dpunkt.verlag Heidelberg 2000.

[18] Mathews, J. H. und K. D. Fink: *Numerical Methods using MATLAB*. Prentice Hall London 1999.

[19] Mohr, R.: *Numerische Methoden in der Technik: ein Lehrbuch mit MATLAB-Routinen*. Vieweg Verlag Braunschweig 1998.

[20] Neundorf, W.: *MATLAB - Teil I: - Vektoren, Matrizen, lineare Gleichungssysteme*. Preprint M 20/99 IfMath der TU Ilmenau, Juli 1999.

[21] Neundorf, W.: *MATLAB - Teil II: - Speicheraspekte, spezielle LGS, SDV, EWP, Graphik, NLG, NLGS*. Preprint M 23/99 TUI, September 1999.

[22] Neundorf, W.: *MATLAB - Teil III: - Komplexe LGS, Interpolation, Splines*. Preprint M 10/00 IfMath der TU Ilmenau, Mai 2000.

[23] Neundorf, W.: *MATLAB - Teil IV: - Approximation, Numerische Intergration*. Preprint M 11/00 IfMath der TU Ilmenau, Mai 2000.

[24] Redfern, D. und C. Campbell: *The MATLAB 5 Handbook*. Springer-Verlag 1998.

Numerik

[25] Deuflhard, P. und H. Hohmann: *Numerische Mathematik*. 1: Eine algorithmisch orientierte Einführung. 3. überarbeitete und erweiterte Auflage, Lehrbuch. Walter de Gruyter Berlin 2002.

[26] Hanke-Bourgeois, M.: *Grundlagen der Numerischen Mathematik und des Wissenschaftlichen Rechnens*. Mathematische Leitfäden. B. G. Teubner GmbH, Stuttgart 2002.

[27] Hermann, M.: *Numerische Mathematik*. R. Oldenbourg Verlag München 2001.

[28] Stoer, J. und R. Bulirsch: *Numerical mathematics 2*. An Introduction - under consideration of lectures by F. L. Bauer. 4. neu bearbeitete und erweiterte Auflage. Springer-Verlag Berlin 2000.

[29] Kiełbasiński, A. und H. Schwetlick: *Numerische lineare Algebra*. Mathematik für Naturwissenschaft und Technik Band 18, DVW, Berlin 1988.

[30] Hackbusch, W.: *Iterative Lösung großer schwach besetzter Gleichungssysteme*. Leitfäden der angewandten Mathematik und Mechanik Band 69. B. G. Teubner Stuttgart 1991.

[31] Maess, G.: *Vorlesungen über numerische Mathematik*. Band 1, 2. Akademie-Verlag Berlin 1984, 1988.

[32] Meister, A.: *Numerik linearer Gleichungssysteme*. Eine Einführung in moderne Verfahren. Friedr. Vieweg & Sohn VG mbH, Braunschweig 1999.

[33] Schwarz, H. R.: *Numerische Mathematik*. B. G. Teubner Stuttgart 1988.

[34] Schaback, R. und H. Werner: *Numerische Mathematik*. Springer-Verlag, Berlin 1992.

[35] Neundorf, W.: *Numerische Mathematik*. Vorlesungen, Übungen, Algorithmen und Programme. Berichte aus der Mathematik. Shaker Verlag Aachen 2002.

[36] Neundorf, W.: *Wissenschaftliches Rechnen - Matrizen und lineare Gleichungssysteme*. Vorlesungsskript IfMath der TU Ilmenau, August 2002.

[37] Neundorf, W.: *Kondition eines Problems und angepaßte Lösungsmethoden*. Preprint M 09/95 IfMath der TU Ilmenau, April 1995.

Anschrift:

Dr. rer. nat. habil. Werner Neundorf
Technische Universität Ilmenau, Institut für Mathematik
PF 10 05 65
D - 98684 Ilmenau

E-mail : werner.neundorf@tu-ilmenau.de
Homepage : http://www.mathematik.tu-ilmenau.de/∼neundorf/index_de.html